Network Resilience Improvement and Evaluation Using Link Additions

By

Mohammed J.F. Alenazi

Submitted to the graduate degree program in Electrical Engineering & Computer Science and the Graduate Faculty of the University of Kansas in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Chairperson: Prof. James P.G. Sterbenz

Prof. Victor S. Frost

Prof. Bo Luo

Prof. Lingjia Liu

Prof. Tyrone E. Duncan

Prof. David Tipper

Prof. Dr. Krzysztof Walkowiak

Date Defended: 21-April-2015

The Dissertation Committee for Mohammed J.F Alenazi
certifies that this is the approved version of the following dissertation:

Network Resilience Improvement and Evaluation Using Link Additions

Chairperson: Prof. James P.G. Sterbenz

Date approved: 21-April-2015

Abstract

Computer networks are getting more involved in providing services for most of our daily life activities related to education, business, health care, social life, and government. Publicly available computer networks are prone to targeted attacks and natural disasters that could disrupt normal operation and services. Building highly resilient networks is an important aspect of their design and implementation. For existing networks, resilience against such challenges can be improved by adding more links. In fact, adding links to form a full mesh yields the most resilient network but it incurs an unfeasibly high cost. In this research, we investigate the resilience improvement of real-world networks via adding a cost-efficient set of links. Adding a set of links to an obtain optimal solution using an exhaustive search is impractical for large networks. Using a greedy algorithm, a feasible solution is obtained by adding a set of links to improve network connectivity by increasing a graph robustness metric such as algebraic connectivity or total graph diversity. We use a graph metric called flow robustness as a measure for network resilience. To evaluate the improved networks, we apply three centrality-based attacks and study their resilience. The flow robustness results of the attacks show that the improved networks are more resilient than the non-improved networks.

Page left intentionally blank.

Acknowledgments

I would like to sincerely thank my advisor, Prof. James P.G. Sterbenz for his support during my studies. I have been inspired by his knowledge within the technical field. I would like to thank the committee members: Prof. Victor S. Frost, Prof. Bo Luo, Prof. Tyrone Duncan, Prof. Lingjia Liu, Prof. David Tipper, and Prof. Dr. Krzysztof Walkowiak for their valuable feedback for improvement of this dissertation.

I would like to thank Egemen Çetinkaya for his guidance from my early stage in the group until now. I also would like to thank Dongsheng Zhang, Yufei Cheng, Truc Anh N. Nguyen, and Siddharth Gangadhar for their suggestions and discussions about research ideas.

I would like to thank the Information and Telecommunication Technology Center (ITTC) network system administrators and the ITTC administrative staff for their support during the development of this dissertation. Michael Hulet, Wesley Mason, Charles Henry, and Paul Calnon have always been helpful in assisting whatever the computing problems I faced.

Last but not least, I would like to thank King Saud University in Saudi Arabia for supporting me during my masters and doctoral studies.

Page left intentionally blank.

Contents

List of Figures

Page left intentionally blank.

List of Tables

Page left intentionally blank.

Chapter 1

Introduction and Motivation

Computer networked applications play an increasingly vital role in supporting a wide range of services. Health care providers and receivers are becoming more dependent on computer networked applications [2]. E-learning is becoming an essential part of academic and professional education [3], and on-line businesses have an increasing number of customers. In 2014, the business-to-consumer (B2C) sales are estimated to be $1.5 trillion while this number is projected to increase in the upcoming years [4]. In addition, most small networks rely on the Internet to communicate with each other. For example, international companies can use the Internet to tunnel their communications among their networks in different cities or countries. The Internet can be divided from a topological point of view to several layers: physical layer, IP, router, PoP (point of presence), and AS (autonomous system) level [5]. The fact that these network services are publicly available makes them prone to targeted attacks. The availability of services depends on the operational state of each level. Each level is susceptible to random failures due to natural events or targeted attacks performed by adversaries.

Computer network resilience is defined as the ability of the network to provide and maintain an acceptable level of service in the face of various faults and challenges to normal operation [6–8]. Since computer networks are susceptible to targeted attacks and

natural disasters that could disrupt its normal operation and services, designing a network with higher resilience can save both lives and money. Increasing the overall resilience of a network can be achieved by the two main mechanisms *redundancy* and *diversity*, which can be applied to several levels of computer networks [7]. In this research, we exploit these two mechanisms while minimizing the cost to add links to physical level networks to improve the overall network resilience. For example, links are added to maximize graph robustness metrics such as algebraic connectivity or path diversity.

The physical topology consists of elements such as fiber and copper cables, point-to-point wireless links, ADMs (add drop multiplexers), cross-connects, and layer-2 switches [9]. From a topological perspective, physical level networks are prone to two types of failures, namely node and link failures. Node failure can be caused by a failure at the operating components at the nodes such as ADMs and layer-2 switches. On the other hand, link failures are mostly caused by fiber cuts, which could be intentional by an adversary or unintentional human error. The node failure is considered to have a higher negative impact than link failures since all links to the failed node also fail. The impact of failing physical nodes or links is different from one network to another based on the graph structure. In a poor physical network design, removing a few nodes or links can have a catastrophic impact on the connectivity of other nodes, which in turn affects the service state negatively. As a result, applications that rely on these network services fail to deliver their services to end users such as banks and hospitals, which could cost money and lives. To cope with this problem, we need to improve network resilience against these node or links attacks. One approach is adding a small additional set of links to the network, these same nodes or links removals can have insignificant effect over network operation and services.

In an attempt to provide *good* network resilience measure, many researchers have proposed graph metrics that assess network robustness against nodes or links removal. For

example, some *classic* graph metrics, such as average node degree, clustering coëficcient, average hop count for shortest paths, radius, and diameter have been used to measure connectivity and robustness. k-connectivity – which indicates the removal of a minimum of k nodes to partition the graph – provides a good robustness measure against node failures. On the other hand, min-cut – which specifies the minimum number of links to partition a graph – provides a good robustness measure against link failures. These two metrics are promising robustness measures; however, the algorithmic complexity for these problems is NP-complete [10], which makes them intractable solutions for large networks.

In this research, we compare the accuracy of several graph properties and graph robustness metrics to predict network resilience against targeted attacks. Using a set of baseline graphs, we calculate graph properties and robustness metrics for each graph to an intuition for how each metric is determined. Then, we present three resilience metrics to measure connectivity against centrality-based node attacks. These metrics calculate the sum of *flow robustness*, which measures the number of remaining reliable flows during each attack [11]. The sum of the flow robustness values, while attacking all nodes, is our measure of network resilience. Using a large set of randomly generated graphs, we check the accuracy of each graph robustness metric to predict graph resilience against centrality-based attacks.

Adding a set of links or nodes to the graphs to optimally maximize a certain graph property is known in the literature as graph augmentation and several problems are proven to be NP-hard [12], even for one objective function such as optimally increasing the algebraic connectivity [13]. Hence, adding objective functions, such as maximizing the algebraic connectivity and minimizing the cost, could be more complex. This complexity increases execution time, which makes these algorithms not practical for some network services such as real-time applications.

Several optimization problems have been approached using a greedy algorithm, which gives a feasible solution to the problem by adding links with local optimality since global optimality is infeasible for large size networks. In this research, we employ this type of algorithm to improve the robustness of a graph by adding a set of links to improve a given robustness graph metric while minimizing the total cost of adding physical links, which could be the proportional to the length of added fiber cables. Using greedy algorithms, we improve the connectivity of a given graph via adding a set of links to maximize a given robustness metric function. We apply our algorithms to real-world networks to generate a set of improved graphs based on presented robustness metrics. Then, we evaluate the non- and improved graphs by applying the three centrality-based attacks to examine their resilience against such attacks. Furthermore, by studying the evaluation results, we show which improvement approach, i.e. maximizing which metric yields better resilient networks.

1.1 Thesis Statement

The design of resilient physical-level networks requires understanding of graph robustness metrics. Moreover, physical-level network resilience evaluation requires understating node and link vulnerability, which can be used to apply worst case scenario attacks. Adding new links to maximize the robustness of a given network can have different outcomes for different robustness metrics. Therefore, our thesis statement is:

> *Network connectivity improvement, via adding a new set of links to maximize a given graph robustness metric under cost constraints, can improve the resilience of the underlying networks against targeted attacks. Determining the best robustness metric can better improve the overall resilience.*

The goal of this dissertation is fivefold:

1. Investigate graph robustness metrics and evaluate their accuracy to predict network resilience against centrality attacks for both synthetic and real service provider networks.

2. Introduce a model to generate physical-level graph based on an unweighted physical-level graph nodes population.

3. Introduce greedy algorithms to improve a given graph using different graph robustness metrics.

4. Evaluate the non- and improved graphs by applying targeted attacks.

5. Compare the results of applying the attacks in terms of their flow robustness and identify the scenarios that yields the best results for each algorithm.

1.2 Proposed Solution

For a given communication network, our objective is to add a set of links so that the network resilience is increased the most against targeted attacks. To fulfill this objective, we first investigate the robustness graph metrics presented in the literature, and from which, we select the most promising metrics based on their accuracy to predicate network resilience against such attacks. Then, we discuss the algorithmic complexity to determine their values since this significantly impacts the algorithm time complexity for large networks.

For each robustness metric, we propose a greedy algorithm that adds links to increase the value for that metric and minimize the total cost, which is assumed to be relative to the length of the added link. For each link added, we show the robustness improvement for

each metric and the cost incurred by adding the link. Then, we apply these algorithms on real-world networks by adding a set of links and generate the improved graphs.

To compare the proposed greedy algorithms, we evaluate the non- and improved graphs by applying several attacks while measuring their flow robustness. In general, there are two types of attack models, namely *random* and *targeted*. Random attacks result in nodes or links are removed from a network based on a random distribution such as the uniform distribution if nodes or links have equal failure likelihood. In targeted attacks, nodes or links are targeted based on their importance. For example, nodes or links with high centrality are attacked first by an intelligent adversary that aims to do the most damage. In this research, we focus on targeted attacks because they cause more destruction to a given network [14] and this is a critical aspect to Future Internet resilience. The flow robustness metric basically measures the number of pair connections alive after the attack and it is discussed in more detail in Section 2.2.1.

By studying the behavior of the improved graphs against the attacks, we can conclude what algorithm performs better in specific scenarios. A comprehensive study is given to provide a cross comparison of applying the proposed algorithms on several real-world networks and their robustness results against attacks. Our research provides an insight for network designers to select what links to add to their networks based on the anticipated attack types.

1.3 Contributions

The main contributions of this dissertation are:

1. Investigate graph robustness metrics and evaluate their accuracy in measuring network resilience against centrality attacks for both synthetic and real service provider

networks.

2. Define flow robustness graph metric for weighted graphs.

3. Model weighted physical-level graph based on nodes population.

4. Design a greedy algorithm to improve algebraic connectivity of a given graph by adding a set of links while minimizing the overall cost for un- and weighted graphs.

5. Design a greedy algorithm to improve total graph diversity of a given graph by adding a set of links while minimizing the overall cost.

6. Design a greedy algorithm to improve balance of the centrality of a given graph by adding a set of links constrained by a given budget while minimizing the overall cost for un- and weighted graphs.

7. Apply the improvement algorithms of real-world graphs and show the improved graphs and their incurred cost.

8. Evaluate and compare the improvement algorithms by applying centrality-based attacks on non- and improved real-world graphs to examine their resilience.

9. For each algorithm, identify the scenarios that yield the best results.

1.4 Relevant Publications

The research presented in this dissertation has resulted in a number of publications, including the following.

Journal articles

10. **Mohammed J.F. Alenazi**, Egemen K. Çetinkaya, and James P.G. Sterbenz, "Cost-Efficient Algebraic Connectivity Optimisation of Backbone Networks," *Elsevier: Optical Switching and Networking.* vol 14, Part 2, August 2014, pp. 107 116.

9. Egemen K. Çetinkaya, **Mohammed J.F. Alenazi**, and James P.G. Sterbenz, "A Comparative Analysis of Geometric Graph Models for Modelling Backbone Networks," *Elsevier: Optical Switching and Networking.* vol 14, Part 2, August 2014, pp. 95 – 106.

8. Egemen K. Çetinkaya, **Mohammed J.F. Alenazi**, Andrew M. Peck, Justin P. Rohrer, and James P.G. Sterbenz, "Multilevel Resilience Analysis of Transportation and Communication Networks," *Telecommunication Systems.*

Peer-reviewed conference proceedings

7. **Mohammed J.F. Alenazi** and James P.G. Sterbenz, "Evaluation and Improvement of Network Resilience against Attacks using Graph Spectral Metrics," submitted to *3rd International Symposium on Resilient Communication Systems*, Philadelphia, August 2015.

6. **Mohammed J.F. Alenazi** and James P.G. Sterbenz, "Comprehensive Comparison and Accuracy of Graph Metrics in Predicting Network Resilience," in *11th International Conference on Design of Reliable Communication Networks (DRCN)*, Kansas City, March 2015.

5. **Mohammed J.F. Alenazi**, Egemen K. Çetinkaya, and James P.G. Sterbenz, "Cost-Efficient Network Improvement to Achieve Maximum Path Diversity," in

Proceedings of the 6th IEEE/IFIP International Workshop on Reliable Networks Design and Modeling (RNDM), Barcelona, Spain, November 2014, pp. 202 – 208.

4. **Mohammed J.F. Alenazi**, Egemen K. Çetinkaya, and James P.G. Sterbenz, "Cost-Constrained and Centrality-Balanced Network Design Improvement," in *Proceedings of the 6th IEEE/IFIP International Workshop on Reliable Networks Design and Modeling (RNDM)*, Barcelona, Spain, November 2014, pp. 194 – 101.

3. **Mohammed J.F. Alenazi**, Egemen K. Çetinkaya, and James P.G. Sterbenz, "Network Design and Optimisation Based on Cost and Algebraic Connectivity," in *Proceedings of the 5th IEEE/IFIP International Workshop on Reliable Networks Design and Modeling (RNDM)*, Almaty, September 2013.

2. Egemen K. Çetinkaya, **Mohammed J.F. Alenazi**, Yufei Cheng, Andrew M. Peck, and James P.G. Sterbenz, "On the Fitness of Geographic Graph Generators for Modelling Physical Level Topologies," in *Proceedings of the 5th IEEE/IFIP International Workshop on Reliable Networks Design and Modeling (RNDM)*, Almaty, September 2013.

1. Egemen K. Çetinkaya, **Mohammed J.F. Alenazi**, Justin P. Rohrer, and James P.G. Sterbenz, "Topology Connectivity Analysis of Internet Infrastructure Using Graph Spectra," in *Proceedings of the 4th IEEE/IFIP International Workshop on Reliable Networks Design and Modeling (RNDM)*, St. Petersburg, October 2012, pp. 752 – 758.

1.5 Additional Publications

Other publications have been resulted from my research during my graduate studies.

5. **Mohammed J.F. Alenazi**, Dongsheng Zhang, Yufei Cheng, and James P.G. Sterbenz, "Epidemic Routing Protocol Implementation in ns-3," *to appear in Workshop on ns-3 (WNS3)*, Spain, Barcelona, March 2015.

4. **Mohammed J.F. Alenazi**, Santosh Ajith Gogi, Dongsheng Zhang, Egemen K. Çetinkaya, Justin P. Rohrer, and James P.G. Sterbenz, "Implementation of Aeronautical Network Protocols," in *Proceedings of the AIAA Infotech@Aerospace Conference*, Boston, MA, August 2013.

3. **Mohammed J.F. Alenazi**, Egemen K. Çetinkaya, Justin P. Rohrer, and James P.G. Sterbenz, "Implementation of the AeroRP and AeroNP Protocols in Python," in *Proceedings of the 48th International Telemetering Conference (ITC)*, San Diego, CA, October 2012.

2. **Mohammed J.F. Alenazi**, Cenk Sahin, Justin P. Rohrer, and James P.G. Sterbenz, "Design Improvement and Implementation of 3D Gauss-Markov Mobility Model" in *Proceedings of the 48th International Telemetering Conference (ITC)*, San Diego, CA, October 2012.

1. **Mohammed J.F. Alenazi**, Santosh Ajith Gogi, Dongsheng Zhang, Egemen K. Çetinkaya, Justin P. Rohrer, and James P.G. Sterbenz, "ANTP Protocol Suite Software Implementation Architecture in Python," in *Proceedings of the 47th International Telemetering Conference (ITC)*, Las Vegas, NV, October 2011.

1.6 Organization

In this chapter we give the overview and motivation performed for this dissertation. The rest of this dissertation is organized as follows: Chapter 2 presents background that

includes graph theory and graph robustness metrics. Graph models are presented in Chapter 3, which includes graph datasets, graph attack models, graph robustness measures. Network design and improvement algorithm are presented in Chapter 4. Chapter 5 presents evaluation of network resilience for our algorithms and a comprehensive comparison. Conclusions and future work are presented in Chapter 6. Finally, a complete set of graph improvement results is presented in Appendix B.

Page left intentionally blank.

Chapter 2

Background and Related Work

This chapter provides background and related work relevant to this dissertation. In Section 2.1, we present the algorithmic complexity associated with resilient network design and show the algorithmic complexity for computing an optimal solution for adding links to a graph to gain a maximum or a minimum objective function by exhaustive search. This section also discusses the greedy algorithm approach to improve the design of a given network. In Section 2.2, we present several graph robustness metrics and their application in network design and robustness evaluation. Finally, we present a framework used to quantify network resilience in the face of challenges and attacks in Section 2.3.

2.1 Network Design and Algorithmic Complexity

Network design and optimization has been studied in the past decades [15, 16] and many problems in this field are considered to be NP-hard [17–20]. Several monographs provide in-depth coverage of the topic [21–24]. In this section, in addition to the discussion of optimal solution via exhaustive search, we briefly present some of the recent work relevant to ours.

2.1.1 Optimal Solution Complexity

Given a graph $G = (N, L)$, $|N|$ is the number of nodes (vertices) and $|L|$ is the number of links (edges); the number of complement links in the graph is $|\bar{G}|$. There are two main generic constraints associated with adding a set of links to a graph. The first constraint is the total cost of adding the links, also called the *budget constraint*. In the worst-case scenario, the optimal solution for the budget constraint requires examining all subsets of the compliment links set. Hence, the algorithm complexity for the budget constraint is defined as:

$$T_{\mathrm{c}}(G) = O(2^{|\bar{L}|}) \tag{2.1}$$

Secondly, The optimal solution with the number of links constrained requires examining all subsets with size n of the compliment links set. Hence, in the worst-case scenario, the algorithm complexity for the number of links constrained is defined as:

$$T_{\mathrm{c}}(G) = O(\binom{|\bar{L}|}{n}) \tag{2.2}$$

Both optimal solutions grow exponentially as the size of the graph increases, which makes them impractical approaches for large-size networks.

2.1.2 Network Design Optimal and Heuristic Algorithms

As the exhaustive search approach fails to be a practical way to provide optimal solutions, heuristic algorithms have been developed to add links between a random pair of nodes and two low-degree nodes to improve the connectivity of graphs [25]. The largest connected component has been measured as an objective function on synthetically generated graphs [25]. Another algorithm has been introduced to improve the robustness of

the networks, in which some links were rewired [26]. Most relevant to our work in this research, an algorithm that minimizes the maximum node betweenness of the graph has been applied to synthetically generated Erdős-Rényi and Barabási-Albert graphs [27,28].

2.1.3 Greedy Algorithms

Greedy algorithms are considered a heuristic approach, which yield feasible solution with local optimum [29]. Given an objective function, the greedy algorithm constructs a feasible solution by selecting one item from a candidate set to maximize or minimize the given objective function. The algorithmic computational complexity for selecting one item is the product of the size of the candidate set and the computational complexity for executing this item. For some problems such as the shortest path, greedy algorithms yield an optimal solution [30]. However, the greedy algorithm does not always guarantee an optimal solution for other problems such as the 0-1 knapsack problem [31]. Since several network design problems have been proven to belong to NP-hard and others are still open-problems, we employ greedy algorithms, which yields feasible solution in polynomial time to improve network resilience via cost-efficient link additions.

2.1.4 Related Work

Several studies have been done to quantify graph robustness against targeted attacks and random failures. Here, we present their work in terms of the proposed robustness metrics and how they have been evaluated. The formula of each metric and its algorithmic complexity are presented in Section 2.2.

Path diversity is a metric that measures disjoint nodes and links between alternative paths between two communicating nodes. The *total graph diversity* (TGD) is the average path diversity among all node pairs [32]. The TGD has shown better accuracy in predicting

survivability of synthetic and real networks when compared to other graph metrics such as clustering coëficcient, average hop count, and betweenness [32]. This metric has been also used to measure network resilience against node and link failures of real-world service provider networks.

Algebraic connectivity has been studied by several researchers [33–35]. It has been shown that algebraic connectivity is more informative and accurate than average node degree when characterizing network resilience [34]. Another study improved synthetically generated Erdős-Rényi random and Barabási-Albert graphs in terms of adding links to the existing topology [33]. On the other hand, one study shows that algebraic connectivity is not tightly related to graph robustness via simulating node and link removals of several random graph types [36]. Furthermore, network optimization can be accomplished by means of rewiring while keeping the number of edges constant [35]. In another research, algebric connectivity has been used to add new links to improve the connectivity of graphs [33].

Weighted spectral distribution (WS) has been introduced to analyze the Internet topology [37]. Another study has been done to compare WS with other robustness metrics against geographic correlated failures and showed that WS is a better measure to evaluate geographically correlated vulnerable links and nodes [38]. In this study, WS has been compared to average node degree, average shortest path, network diameter, network criticality, and algebraic connectivity to measure network survivability against correlated failures in real-world weighted networks. The results have shown that WS is the most versatile measure in evaluating such failures.

Natural connectivity is a spectral graph metric that has been compared to algebraic connectivity using a set of structural and random graphs to examine robustness against node and link removals [39]. In this research, natural connectivity has been compared

other measures such as edge connectivity and algebraic connectivity in the presence of link failures of real-world networks. The results have shown that the natural connectivity measures connectivity changes more precisely than algebraic connectivity.

Network criticality is a spectral graph metric that measures the robustness of a network against topological changes [40]. A smaller value of network criticality means higher network robustness. Furthermore, this metric has been compared to algebraic connectivity, average node degree, and average node betweenness. However, this study concluded that there is no unique graph metric that can capture robustness and connectivity [41].

The *spectral gap* is also a spectral graph metric that has been used to measure the robustness of the graph against targeted attacks [42]. A small spectral gap value indicates a smaller number of articulation points that might partition the network once a node or a link is removed [43].

Effective graph resistance is a spectral graph metric that measures the robustness of network against node or link removals [44]. This metric has been compared to algebraic connectivity in terms of measuring the connectivity of several random types and real-world networks. The structure of this metric is similar to network criticality since both metrics use the Moore-Penrose inverse of Laplacian matrix of the given graph [40].

2.2 Graph Robustness Metrics

In this section, we present and discuss several graph, node, and link metrics associated with graph robustness.

2.2.1 Flow Robustness

Flow robustness is a graph metric that measures the ratio of the number of reliable flows to the number of total flows in the network [11, 32]. A flow is considered *reliable* if at least one of its paths remains unbroken by the link or node failures. The number of total flows is the maximum number of flows, which is $n(n-1)/2$ flows for n nodes. This metric captures the ability for the network nodes to communicate with each other after a node or link is removed. The range for flow robustness values is $[0, 1]$ where 1 indicates that all the nodes can communicate with each other and 0 means there is no node-pair communication in the whole network i.e, there are no links in the graph. The flow robustness metric can be used to model packet delivery ratio (PDR), where packet buffering is disabled. For example, the flow robustness value can represent the upper bound value for the constant bit rate (CBR) traffic model with no buffering mechanism. To calculate flow robustness, let $G = (N, L)$ be the graph representing the given network. Let $\{C_i; 1 < i < k\}$ be the set of components in graph G. The flow robustness FR is computed using:

$$\mathrm{FR}(G) = \frac{\sum_{i=1}^{k} |C_i|(|C_i| - 1)}{|V|(|V| - 1)}, \qquad 0 \leq \mathrm{FR} \leq 1 \tag{2.3}$$

The algorithmic complexity to calculate FR is heavily dependent on the complexity to find the number of components in a given graph, which is $O(|N| + |L|)$ [45]. Since the maximum value for k is $|N|$, the worst-case complexity for the summations is also $|N|$. Thus, the algorithmic complexity for calculating the flow robustness is $O(|N|+|L|+|N|)$, which can be simplified to $O(|N| + |L|)$.

To illustrate how the flow robustness captures the connectivity of graphs, we compute it for six graphs with four nodes and different connectivity as shown in Figure 2.1. In

this example, the first two graphs: G_1 and G_2 have one component each with a different number of links. G_3 and G_4 have two components each with the same number of links. G_5 has three components with one link and G_6 has four components and no links. For each graph in Figure 2.1, we calculate the flow robustness using Equation 2.3. The flow robustness values for each graph are presented in Table 2.1. Even though G_1 and G_2 have different number of links, they have the same flow robustness value of 1 because there exists a reliable flow between each pair of nodes. G_3 and G_4 have the same number of nodes, links, and components; however, they have a different flow robustness values. G_3 has a larger flow robustness value than G_4 since it has more reliable flows. G_5 has a relatively low flow robustness value since it just has one reliable flow. Finally, G_6 has a 0 value of flow robustness because there are no reliable flows in the graph.

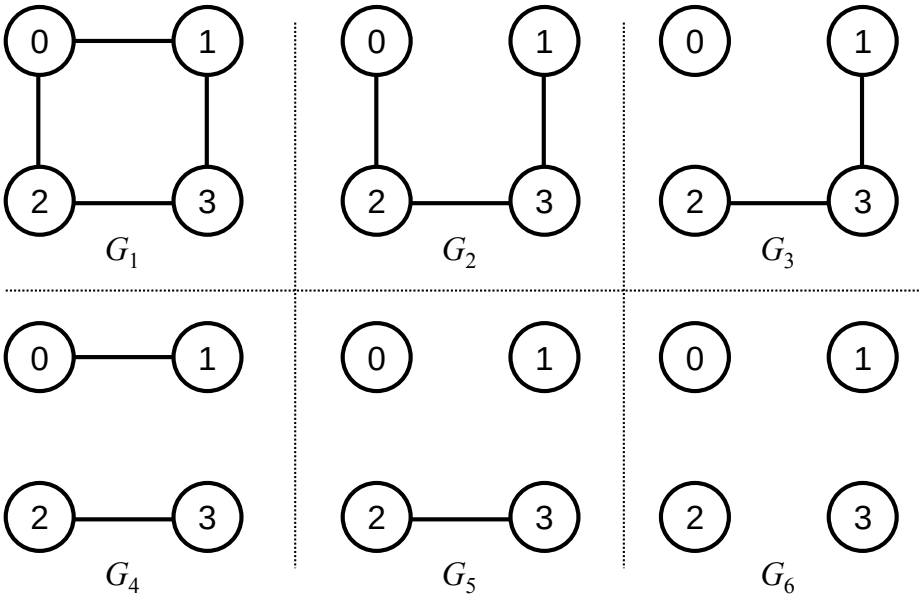

Figure 2.1: Unweighed flow robustness calculation example

2.2.2 Graph Centrality Metrics

For a given graph $G = (N, L)$ with a set of nodes N and links L, centrality metrics show the importance of a link or a node to the graph. Since the link or node importance

Table 2.1: Flow robustness values in six graphs

Graph	**Nodes**	**Links**	**Components**	**Flow Robustness** (FR)
G_1	4	4	1	1.00
G_2	4	3	1	1.00
G_3	4	2	2	0.50
G_4	4	2	2	0.33
G_5	4	1	3	0.17
G_6	4	0	4	0.00

varies from one application to another, several metrics have been introduced as indicators to identify central nodes based on the need of the application. Here, we focus on the commonly used metrics: betweenness, closeness, and degree centrality [46–48].

Shortest path

The shortest path is not a centrality metric itself but since it is used as the basis of several centrality metrics, we present its definition for both unweighted and weighted graphs. For unweighted graphs, the shortest path of nodes i and j is the path with minimum number of hops. It is denoted as:

$$d_{ij} = \min\{x_{ih} + ... + x_{kj}\} \tag{2.4}$$

where $x_{ih} + ... + x_{kj}$ represents all possible paths between node i and node j [49].

For weighted graphs, the shortest path of two nodes i and j is the path from nodes i and j with the minimum total cost. For communication networks, if the weight represents a connectivity measure, then the cost is the inverse of the weighted because paths with higher connectivity are preferred. As a result, the path with minimum total cost is in fact the path with high total connectivity [49, 50]. Hence, the shortest path of two node

i and node j in weighted graphs is defined as:

$$d_{ij}^{w} = \min\{\frac{1}{w_{ih}} + ... + \frac{1}{w_{kj}}\} \tag{2.5}$$

Since the sum of the total cost can dominate the value of the weighted shortest path, another approach introduces a tuning parameter to control the impact of either total cost or number of hops in the weighted shortest path [49]. The equation to determine the shortest path is defined as follows:

$$d_{ij}^{w\alpha} = \min\{\frac{1}{(w_{ih})^{\alpha}} + ... + \frac{1}{(w_{kj})^{\alpha}}\} \tag{2.6}$$

where α is a positive tuning factor that can be set based on application. When $\alpha = 0$ the weight impact is completely ignored and the equation becomes identical to Equation 2.4. However, when $\alpha = 1$ the weight impacts dominates and the equation is identical to Equation 2.5.

Some other common graph metrics such as *eccentricity*, *radius*, and *diameter* provide statistical graph values of all node-pair shortest paths. The *eccentricity* $\varepsilon(n)$ is the longest of the shortest paths between node n and every other node. The graph *radius* R(G) is the *shortest* of the shortest paths of graph G. The graph *diameter* D(G) is the *longest* of the shortest paths of graph G.

The algorithmic complexity to determine the shortest path from one node to all other nodes using Dijkstra's algorithm is $O(|N|^2)$ [51]. To determine the shortest paths among all pairs in the graph, the Floyd-Warshall algorithm has an algorithmic complexity of $O(|N|^3)$ [52].

Degree

One of the simplest and yet most commonly used centrality metric is degree centrality, defined as the number of link incidents to a node and can be viewed as the importance of connectivity of a node [53]. The degree is a local centrality metric since it depends only on the number of links locally connected. In communication networks, nodes with a high degree are considered to be more important than nodes with a lower degree because they they provide connectivity to many other links.

For weighted graphs, one approach defines the weighted node degree as the sum of link weights connected to that node [50, 54]. However, this approach can be ambiguous in some cases; for example, if we have a node with just one link of weight 5 and another node with three links of weight 1. The weighted degree for the first node is 5 and for the second node is 3. Even though the second node has more attached links (higher weighted degree), the weighted degree is smaller. Another approach tries to tackle this ambiguity by adding a tuning factor α to control the impact of weights on calculating the weighted degree [49]. Hence, the weighted degree of node i is formally defined as:

$$C_{\mathrm{D}}^{w\alpha}(i) = k_i^{(1-\alpha)} \left(\sum_{j=1}^{N} w_{ij} \right)^{\alpha} \tag{2.7}$$

where α is a positive tuning factor that can be set based on application, k_i is the unweighted degree of node i, and w_{ij} represents the link weight between n_i and n_j [49]. Assigning $\alpha = 0$ ignores the effect of cost and yields the unweighted degree equation. On the other hand, assigning $\alpha = 1$ ignores the effect of unweighted degree and yields the sum of link weights for the given node. The algorithmic complexity to determine a node degree is $O(|N|)$.

Assortativity

Assortativity As(G) is a graph metrics that measures the degree similarity among adjacent nodes for a given graph G [55]. For example, in a uniform-degree graph the assortativity is 1. The assortativity is defined as:

$$\mathrm{As}(G) = \frac{m^{-1}\sum_i j_i k_i - [m^{-1}\sum_i \frac{1}{2}(j_i + k_i)]^2}{m^{-1}\sum_i \frac{1}{2}(j_i^2 + k_i^2) - [m^{-1}\sum_i \frac{1}{2}(j_i + k_i)]^2} \tag{2.8}$$

where j_i, k_i are the degrees of the nodes at the ith link and m is the number of links.

Betweenness

Betweenness is centrality graph metric that can used for both nodes and links. Node betweenness is defined as the number of shortest paths through a node. On the other hand, link betweenness is defined the number of the shortest paths through a link. Betweenness is considered to have global significance since the betweenness value is impacted by the whole structure of the graph [56]. In communication networks, nodes with high betweenness exhibit more importance as they carry more packet flows and these failures results in significant traffic disruption and rerouting.

$$C_{\mathrm{B}}(v) = \sum_i \sum_j \frac{g_{ij}(v)}{g_{ij}} \quad , \quad v \neq i \neq j \tag{2.9}$$

For weighed graphs, the number of shortest paths g_{ij} is computed using the weighted shortest path presented in Equation 2.5 and denoted as g_{ij}^w. The weighted betweenness is defined as:

$$C_{\mathrm{B}}^w(v) = \sum_i \sum_j \frac{g_{ij}^w(v)}{g_{ij}^w} \quad , \quad v \neq i \neq j \tag{2.10}$$

To control the impact of cost, the number of shortest paths g_{ij}^w is computed using the weighted shortest path with α tuning parameter presented in Equation 2.6 and denoted

as g_{ij}^{w}. The α tuned weighted betweenness is defined as:

$$C_{\mathrm{B}}^{w\alpha}(v) = \sum_{i} \sum_{j} \frac{g_{ij}^{w\alpha}(v)}{g_{ij}^{w\alpha}} \quad , \quad v \neq i \neq j \tag{2.11}$$

where α is a positive tuning factor that can be set based on application [49]. The algorithmic complexity to determine node betweenness is $O(|N||L|)$ [57].

Closeness

For unweighted graphs, closeness is a node centrality metric that measures the mean distance from the node to other nodes [53, 58]. It is defined as:

$$C_i = (\sum_{j=1}^{|V|} d_{ij})^{-1} \tag{2.12}$$

d_{ij} is the distance between nodes i and j presented in Equation 2.4.

For weighted graphs, weighted closeness of node i is computed using the weighted shortest path presented in Equation 2.5 and defined as:

$$C_i^{w} = (\sum_{j=1}^{|V|} d_{ij}^{w})^{-1} \tag{2.13}$$

To control the impact of the weight, the weighted closeness of node i is computed using the weighted shortest path with α tuning parameter presented in Equation 2.6 and defined as:

$$C_i^{w\alpha} = (\sum_{j=1}^{|V|} d_{ij}^{w\alpha})^{-1} \tag{2.14}$$

In communication networks, closeness indicates the efficiency of a message's diffusion in a network. The algorithmic complexity to determine the closeness for a given graph is $O(|N|^3)$ [59].

2.2.3 Graph Spectra Robustness Metrics

Let $G = (N, L)$ be an unweighted, undirected graph with $|N|$ nodes and $|L|$ links. The topology of G can be represented by an adjacency matrix, incidence matrix, Laplacian matrix, or normalized Laplacian matrix [60, 61]. Given a symmetric adjacency matrix, $A(G)$, with no self-loops where $a_{ii} = 0$, $a_{ij} = a_{ji} = 1$ if there is a link between $\{n_i,n_j\}$, and $a_{ij} = a_{ji} = 0$ if there is no link between $\{n_i,n_j\}$. Then, the Laplacian matrix of G is the difference between the diagonal matrix and the adjacency matrix; $L(G) = D(G) - A(G)$. The diagonal matrix, $D(G)$, represents node degrees, $d_{ii} = \deg(v_i)$. The normalized Laplacian matrix $\mathcal{L}(G)$ can be represented:

$$\mathcal{L}(G)(i,j) = \begin{cases} 1, & \text{if } i = j \text{ and } d_i \neq 0 \\ -\dfrac{1}{\sqrt{d_i d_j}}, & \text{if } v_i \text{ and } v_j \text{ are adjacent} \\ 0, & \text{otherwise} \end{cases}$$

Spectral graph theory studies the relationship between graph structural properties and *eigenvalues* and *eigenvector* of their adjacency, incidence, and Laplacian matrices. This topic has been extensively covered in several monographs [60–64]. For weighted graphs, using the weighted degree definition presented in Equation 2.7, the weighted graph spectra can be computed using the same method for the unweighted graphs [65].

Algebraic connectivity

Algebraic connectivity, denoted as $a(G) = \lambda_2$ has been studied by several researchers [33–35]. It has been shown that algebraic connectivity is more informative and accurate than average node degree when characterizing network resilience [34]. Moreover, we have shown algebraic connectivity [9, 66, 67] and diversity [32] are *predictive* of flow robustness of graphs. Three synthetically generated topologies (i.e. Watts-Strogatz *small-world*,

Gilbert *random*, and Barabási-Albert *scale-free*) have been improved using link rewiring in which the objective is to increase the algebraic connectivity [35]. It has been shown that algebraic connectivity increases the most if links are rewired between weakly connected nodes. Another study improved synthetically generated random Erdős-Rényi and Barabási-Albert graphs in terms of adding links to the existing topology [33].

Spectral gap

The *spectral gap*, denoted as $\Delta\lambda = \lambda_n - \lambda_{n-1}$, is the difference between the largest and second largest eigenvalues of the adjacency matrix. The spectral gap has been used to show the robustness of the graph against targeted attacks [42]. A small spectral gap indicates articulation points that might partition the network once a node or a link is removed [43].

Natural connectivity

Natural connectivity, denoted as $\bar{\lambda}$, is a scaled average eigenvalue of the graph adjacency matrix [39]. A larger value of $\bar{\lambda}$ indicates, more robustness to link or nodes removals. Natural connectivity has been compared to algebraic connectivity using a set of structural and random graphs to predict robustness against node and link removals [39]. The value of $\bar{\lambda}$ is calculated as follows:

$$\bar{\lambda} = \ln [\frac{1}{n} \sum_{j=1}^{n} e^{\lambda_j}] \tag{2.15}$$

where λ_j is the jth eigenvalue of the adjacency matrix.

Weighted spectrum

Weighted spectral distribution, denoted as WS, has been introduced to analyze Internet topology [37]. The value of WS is calculated as follows:

$$\mathrm{WS}(G, N) = \sum_{i=1}^{n} (1 - \lambda_i)^N \tag{2.16}$$

where λ_i is the ith eigenvalue of the normalized Laplacian matrices and N is the number of cycles being measured [37]. In this research, we use $N = 4$ since it is related to the number of disjoint paths [37,38]. A study has been conducted to compare WS with other robustness metrics against geographic correlated failures and showed that WS is a better measure to evaluate geographically correlated vulnerable links and nodes [38].

Network criticality

Network criticality, denoted as $\hat{\tau}$, is a graph metric that measures the robustness of network against topological changes [40]. A smaller value of $\hat{\tau}$ indicates higher network robustness. Moreover, this metric has been compared with other robustness metrics [41]. We note that this metric is also called *total resistance distance* [68]. The value of $\hat{\tau}$ is calculated as follows:

$$\hat{\tau} = \frac{2}{n-1} \mathrm{Trace}(\mathcal{L}^+) \tag{2.17}$$

where n is the number of nodes in a given graph, $\mathrm{Trace}(\mathcal{L}^+)$ is the trace of the Moore-Penrose inverse of Laplacian matrix of the given graph [40].

Effective graph resistance

Effective graph resistance, denoted as R_G, is a graph metric that measures the robustness of network against node or link removals. The value of R_G is calculated as follows:

$$R_G = N \sum_{i=2}^{N} \frac{1}{\lambda_i} \tag{2.18}$$

A normalized version is called *effective graph conductance* C^*, which is defined as:

$$C^* = \frac{N-1}{R_G} \tag{2.19}$$

where the values of C^* lie in the interval $[0, 1]$.

Graph spectra algorithmic computational complexity

Given a graph with n nodes and m links, the corresponding adjacency matrix, incidence matrix, Laplacian matrix, and normalized Laplacian matrix are of size $n \times n$. Using the worst-case algorithmic analysis, big-O, the computational complexity for determining the eigenvalues of the graph is $O(n^2)$. Thus, the algorithm complexity of calculating $\hat{\tau}$ is $O(mn^{\frac{3}{2}} \log(n))$ [40], where m is the number of links.

2.2.4 Path Diversity

Algorithms and mechanisms are necessary to defend networks and to make them resilient against challenges [7]. One such mechanism is *diversity* and it has been the subject of a number the published works in the field of network resilience. Diversity is used to enhance bandwidth, delay, and loss rate of media streaming applications [69]. Path diversity is used in the optical domain to route around failed nodes or to split traffic for a better utilization of network resources [70]. Diverse routing is necessary for multihoming to improve the service delivery of provider networks [71, 72]. While the path diversity of a

graph is essential for survivable design, it can be improved by addition of links in a given graph using an improvement algorithm.

A path between a source s and a destination d is the set of nodes and links that form a loop-free connection. Diverse paths between node pairs strengthen the ability of a network to withstand attacks and correlated failures. If the alternative paths have no common nodes or links they are disjoint, and if there are common network nodes or links, they are partially disjoint. Path diversity has been studied from a topological perspective [73–75], as well as in terms of multipath routing [70,72,76–82], and multipath transport [32,83]. Furthermore, multipath routing has been studied to improve the QoS (Quality of Service) of networks [78,79], the resilience of interdomain routing [76,80], and the survivability in optical networks [70,81,82]. Moreover, finding disjoint paths between a node pair is considered to be a NP-complete problem [84,85]. Next, we explain path diversity and the path diversity of a graph [11,32].

Path diversity definition

Given a shortest path and an alternative path between two nodes in a graph, the path diversity of the alternative is defined as the ratio of the number of disjoint elements (nodes *and* links) between the shortest path and alternative path to the number of elements in the shortest path. Given a (source s, destination d) node pair, a path P between them is a set containing all links L and all intermediate nodes N [32],

$$P = L \cup N \tag{2.20}$$

and the *length* of this path $|P|$ is the combined total number of elements in L and N. Let the shortest path between a given (s, d) pair be P_0. Then, for any other path P_k

between the same source and destination, the definition of the diversity function [32, 76] $D(P_k)$ with respect to P_0 as:

$$D(P_k) = 1 - \frac{|P_k \cap P_0|}{|P_0|} \tag{2.21}$$

The path diversity has a value of 1 if P_k and P_0 are completely disjoint and a value of 0 if P_k and P_0 are identical. This measure captures the diversity with respect to both nodes and links on alternative paths [32]. As an example, we illustrate finding the path diversity of the paths between node 0 and 2 in the graph shown in Figure 2.2.

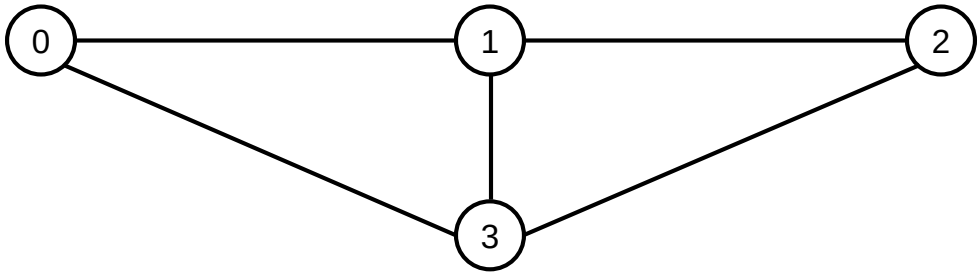

Figure 2.2: Path definition example

There are four possible paths namely: $P_0 = [0, 1, 2]$, $P_1 = [0, 3, 2]$, $P_2 = [0, 1, 3, 2]$, and $P_3 = [0, 3, 1, 2]$. The shortest path in this graph is P_0 so its path diversity is zero. However, to calculate the path diversity for P_1, we first convert the paths to the path element sets. For P_0, the path elements set is $\{(0, 1), 1, (1, 2)\}$. The tuples $(0, 1)$ and $(1, 2)$ represent the links while the element 1 represents the node 1 in the path. We do not include the source and destination nodes in order to have a path diversity of 1 when the two paths are fully disjoint. Using the same method, the path elements set for P_1 is $\{(0, 3), 3, (3, 2)\}$. Using Equation 2.21, $D(P_1) = 1 - \frac{0}{3} = 1$. To find the path diversity for P_2, the path elements set for P_2 is $\{(0, 1), 1, (1, 3), 3, (3, 2)\}$. Again, using Equation 2.21, $D(P_2) = 1 - \frac{2}{3} = \frac{1}{3}$. Finally the path diversity of P_3, using the same procedure is

$D(P_3) = 1 - \frac{2}{3} = \frac{1}{3}$. We note that converting a path to its elements method assumes directed graphs. In this research, since we study the path diversity of undirected graphs, we modify the method to work with undirected graphs. Thus, to construct a path to element set, we start with the same method for the directed graph. Then, for each link (a, b) in the resulting set, we add (b, a) to the set.

Capturing path diversity of a graph

TGD (total graph diversity) is the average of the EPD (effective path diversity) values of all node pairs in a given graph [32] and TGD measures the path diversity of a graph as a single value. EPD is the normalized sum of path diversities for a selected set of paths connecting a node pair (s, d). First, we find the k diverse paths using the algorithm presented in Section 4.2.1. Then, we remove zero diversity paths from the list of returned paths because they do not add any additional diversity. The returned diverse path is denoted as $P_{s,d} = \{P_1, P_2, \ ... \ , P_m\}$, where $m \leq k$, since-zero diversity paths are removed from the set. To calculate EPD, we use the exponential function:

$$\text{EPD} = 1 - e^{-\lambda k_{sd}} \tag{2.22}$$

where k_{sd} is the sum of all non-zero diversity paths defined as:

$$k_{sd} = \sum_{i=1}^{m} D(P_i) \tag{2.23}$$

where $D(P_i)$ is the non-zero path diversity of the i-th path with respect to the P_0. In Equation 2.22, λ is an experimentally determined constant that scales the impact of k_{sd} based on the utility of this added diversity [32]. For a given pair of nodes, the range of EPD is between $[0, 1)$ where 0 means that there is no diversity in between the two nodes

as there are no alternative paths connecting the pair. When the EPD approaches 1, it means that the network provides high path diversity [32].

2.3 Analytical Resilience Framework

An evaluation framework has been developed to quantify network resilience using a two-dimensional state space as the network is challenged by failures, attacks, or large-scale disasters [1, 7, 86]. We will use this framework to show the resilience of a given network during targeted attacks. Resilience $\mathbb{R}$ is a function of network operational state $\mathbb{N}$ and service state $\mathbb{P}$. First, let $\mathbb{N}_{\mathcal{S}}$ be a set of operational states, $\mathbb{N}_{\mathcal{S}} = \{\mathbb{N}_1, \ldots, \mathbb{N}_k\}$. The framework divides the operational state into three regions: *normal operation*, *partially degraded*, and *severely degraded*. Second, the framework characterizes the service provided at a given network layer. Let $\mathbb{P}_{\mathcal{S}}$ be a set of service states, $\mathbb{P}_{\mathcal{S}} = \{\mathbb{P}_1, \ldots, \mathbb{P}_k\}$. Similarly, the service space $\mathbb{P}$ is divided into *acceptable*, *impaired*, and *unacceptable* regions [1,86,87]

Under normal conditions, the network continues to operate in a given state corresponding to normal operational and service states. When the network is under attack, which causes a large degradation in the operational state, the service may also be impaired below the acceptable service specification. A significant change in either dimension leads to a *network state transition*. The framework formulates that challenges in the form of adverse events transform the network from one *state* to another based on the severity of the event [1].

Network resilience can be evaluated in terms of the various network state transitions under the presence of challenges. The network operational space $\mathbb{N}$ is divided into *normal operation*, *partially degraded*, and *severely degraded* regions as shown in Figure 2.3. Similarly, the service space $\mathbb{P}$ is divided into *acceptable*, *impaired*, and *unacceptable* regions. The framework then quantifies the resilience $\mathbb{R}$ for a particular scenario as the area

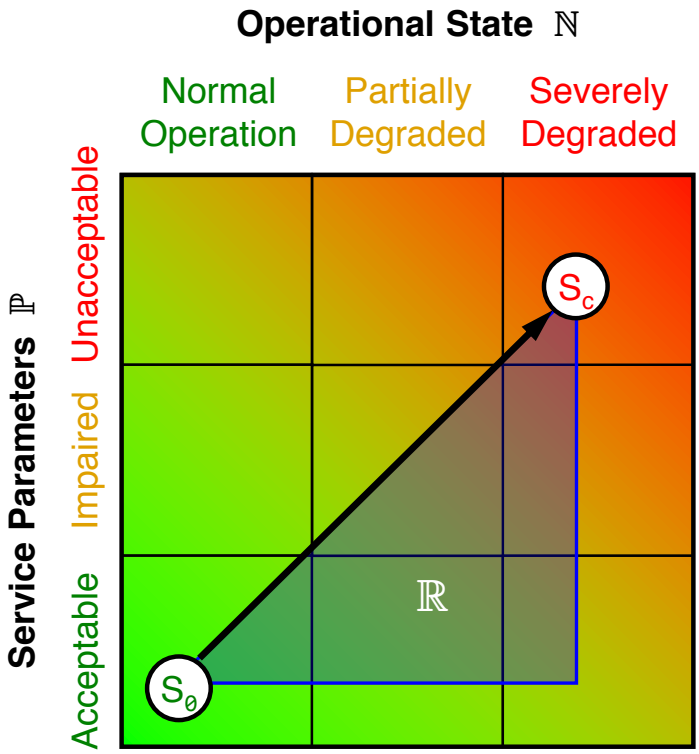

Figure 2.3: Resilience $\mathbb{R}$ measured in state space [1]

under the resilience trajectory, shown by the shaded triangular area under the $S_0 \rightarrow S_c$ trajectory in Figure 2.3 [1].

2.4 Summary

We presented the network design and improvement algorithmic complexity and showed the complexity of computing an optimal solution by an exhaustive search. In addition, we discussed using a greedy algorithm for network design. Then, we discussed several graph robustness metrics for un- and weighted graphs and discussed their applications in network design and robustness evaluation. Finally, we presented a framework used to quantify network resilience in the face of challenges and attacks.

Page left intentionally blank.

Chapter 3

Graph Models

In this chapter, we present our dataset, our centrality-balanced robustness metrics, our weighted flow robustness metric, graph attack models, and how network resilience is measured. Our dataset includes baseline graphs, random graphs, and real-world graphs. We present our graph robustness metrics that capture how a given graph's centrality is balanced. We then present our weighted flow robustness and show it is used to capture network resilience. After that, we show several centrality-based attack models and how they are used to measure network resilience and we present our model to convert unweighted graphs to weighted graphs using node populations and link betweenness. Finally, we present how we use flow robustness to measure network resilience against centrality attacks.

3.1 Graph Dataset

In this section, we present our datasets, which consist of four categories: *baseline*, *random*, *unweighted* real-world, and *weighted* real-world graphs. For each category, we present several graph types and discuss their properties. In all our models, we use simple undirected graphs.

3.1.1 Baseline Graphs

We select a set of graphs with known structures to give some intuition of each graph metric during the evaluation process. This set includes: *full-mesh*, *wheel*, *grid*, *torus*, *ladder*, *ring*, *barbell*, *linear*, *binary-tree*, and *star* graphs. A *full-mesh* graph, also called *complete*, has a link between every node pair. A *star* graph has one node designated as the root and a set of other nodes, while there is a link between the root and every other node. A *wheel* graph is a star graph with a link connecting every adjacent leaf. A *grid* graph has a $m \times n$ nodes placed in a grid form with m rows and n columns. In this graph, there is a link connecting every adjacent vertical and horizontal node pair. A *torus* graph is a grid graph with a link connecting every left and right node in each row and a link connecting every top and bottom node in each column. A *ladder* graph is a special case of the grid graph such that n is always 2. A *linear* graph, also called a *path*, is a set of nodes placed as a line in which there is a link connecting every adjacent node pair. A *ring* graph, also called a *circle*, is a linear graph with the end nodes connected by a link. A *binary-tree* graph has one node designated as a root with two children nodes. Each child has at most two children with a hight h defining the length of the shortest path between the root and the leaves. A *barbell* has two full mesh graphs and a link connecting them. A set of examples for the baseline graphs are shown in Figure 3.1.

3.1.2 Random Graphs

In this section, we present three random graph models to generate our dataset for robustness metrics evaluation.

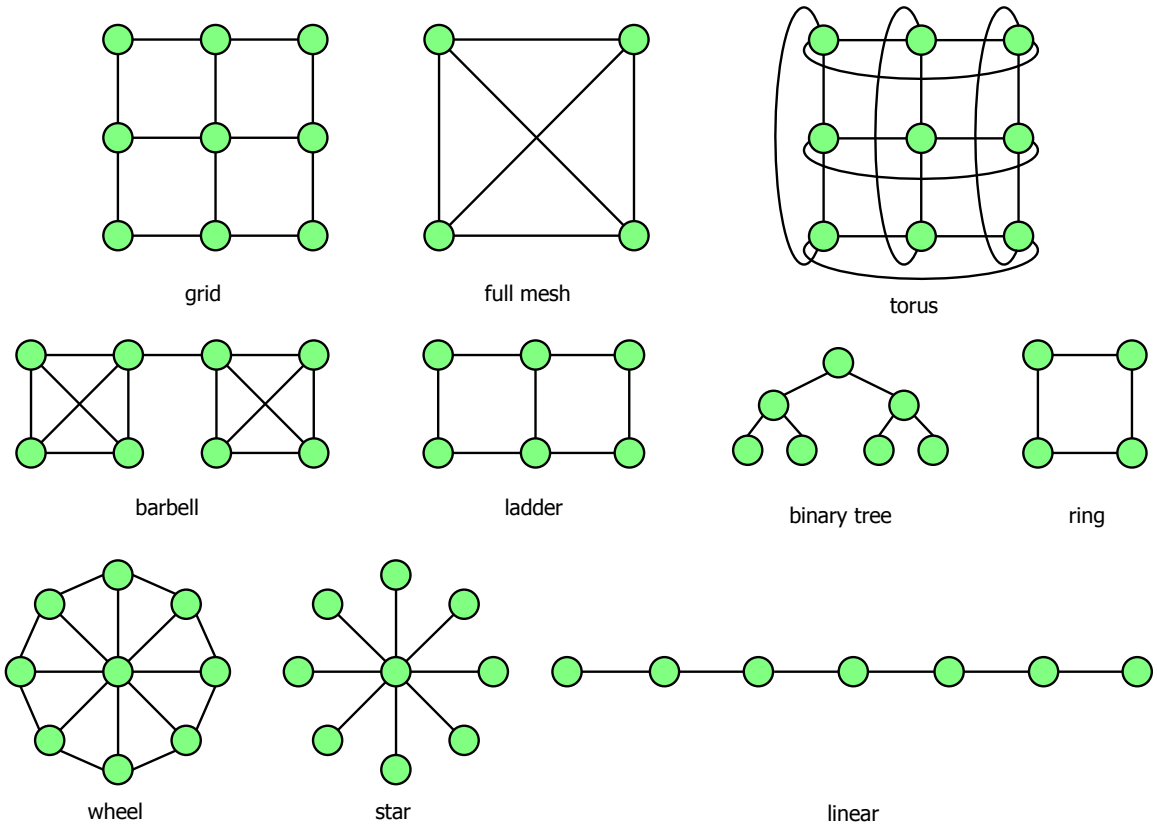

Figure 3.1: Several examples of baseline graphs

Table 3.1: Properties of baseline graphs

Graph	Nodes	Links	Avg. Node Degree	Radius	Diameter	Avg. Shortest Path
full mesh	10	45	9.00	1	1	1.00
torus	9	18	4.00	2	2	1.50
wheel	10	18	3.60	1	2	1.60
barbell	12	17	2.83	4	7	3.48
binary tree	15	14	1.87	3	6	3.50
ladder	10	13	2.60	3	5	2.33
grid	9	12	2.67	2	4	2.00
ring	10	10	2.00	5	5	2.78
linear	10	9	1.80	5	9	3.67
star	10	9	1.80	1	2	1.80

Gilbert graphs

The Gilbert random graph model is one of the earliest models to construct random graphs [88]. Given a number of nodes n and connectivity probability p, the random graph model $\mathrm{G}(n, p)$ constructs a graph with n nodes and m links are connected with probability p. Another similar graph model is the Erdős-Rényi (ER). The random graph model $\mathrm{ER}(n, m)$ generates a graph with n nodes and randomly connected m links.

Waxman graphs

The Waxman model provides a probabilistic way of connecting nodes in a graph [89]. Given two nodes $\{u, v\}$ with a Euclidean distance $d(u, v)$ between them, the probability of connecting these two nodes is:

$$P(u, v) = \beta e^{\frac{-d(u,v)}{L\alpha}} \tag{3.1}$$

where $\beta, \alpha \in (0, 1]$ and L is the maximum distance between any two nodes. Increasing β increases the link density and a large value of α corresponds to a high ratio of long links to short links. In this research, the Waxman model node locations are uniformly distributed. It has been shown that Waxman graphs exhibit mesh-like properties of logical-level networks [90].

Gabriel graphs

Gabriel graphs are useful in modeling graphs with geographic connectivity that resemble grids [91, 92]. In a Gabriel graph, two nodes are connected directly if and only if there are no other nodes that fall inside the circle whose diameter is given by the line segment joining the two nodes. The node locations are generated randomly using a uniform

distribution with a range of $[0, 1]$ for both x-axis and y-axis. It has been shown that Gabriel graphs exhibit grid-like properties of physical-level networks [90].

3.1.3 Unweighted Real-World Networks

For our real-world networks, we use five physical ISP networks. We make use of the publicly available: AT&T [93], Sprint [94] Internet2 [95], Level 3 [96], and CORONET [97,98], fiber-level topologies. CORONET is a synthetic fiber topology designed to be representative of service provider fiber deployments. Moreover, we present some well-known graph metrics for each graph to provide an insight of the graph properties, including number of nodes, number of links, degree of connectivity, radius, diameter, and average hop count as shown in Table 3.2. A set of all the unweighted maps is shown in Figure A.1.

Table 3.2: Physical graph properties of three service provider networks

Graph	Nodes	Links	Avg. Node Degree	Radius	Diameter	Avg. Shortest Path
AT&T	361	466	2.58	19	37	13.57
Sprint	263	311	2.37	19	37	14.78
CORONET	75	99	2.64	9	17	6.45
Internet2	57	65	2.28	8	14	6.69
Level 3	99	132	2.67	10	19	7.65

3.1.4 Weighted Real-World Networks

We use several weighted real-world graphs for physical networks from *The Internet Topology Zoo* [99]. The properties of these topologies are shown in Table 3.3. KAREN or REANNZ (Research and Education Advanced Network New Zealand Ltd) is the Crown-owned company that owns and operates a high-speed, unrestricted broadband network for New Zealand education, research and innovation communities [100]. InternetMCI represents the physical topology of MCI Communications Corp, which later merged with

WorldCom [101]. CARNet (Croatian Academic and Research Network) is a public institution that today operates under the Croatian ministry of science, education and sports in the field of information and communication technologies [102]. GÉANT is the European research and education network that interconnects several institutions across Europe, supporting research in areas such as energy, the environment, space, and medicine [103]. A set of all the weighted maps is shown in Figure A.2.

Table 3.3: Weighted graph properties of five service provider networks

Graph	Nodes	Links	Avg. Node Degree	Radius	Diameter	Avg. Hopcont
KAREN	25	28	2.24	4	7	3.42
InternetMCI	19	33	3.47	3	4	2.39
CARNet	44	43	1.95	3	6	2.99
GÉANT	37	56	3.03	4	7	3.46

3.1.5 Physical Level Networks Dataset Constraint

Each improvement algorithm selects new links from a candidate set. All possible candidate links are located in the input graph complement. However, this set may contain very long links that are not practical to be added to a physical graph. For example, adding a physical fiber link between Los Angeles and Boston is unlikely to be feasible for providers given the high cost associated by adding this link. In our implementation, the candidate set only contains links that are not longer than a specified threshold value. Therefore, this raises the question of what the best threshold value should be. In this research, we choose the current maximum length link in the input graph to be the threshold for removing long links from the graph complement links. We assume this value gives a good indicator for the maximum link length a provider can afford.

3.1.6 Population-Based Weighted Graph Model

There are many real-world and synthetic physical level networks available as datasets for researchers [104]. However, most of these networks are modeled using unweighted graphs because most providers are unwilling to disclose their network topological properties in order to maintain security and competitiveness. While topologies can be inferred or mined from public documents, capacity information is much harder to obtain. A few weighted real-world graphs are available for educational institutes presented in Section 3.1.4. For the synthetic graphs, most graph generating algorithms deal with determining the link existing among a given number of nodes. Modeling physical level networks using weighted graphs provides a more accurate description of their behavior against random and targeted attacks. Thus, in this section, we introduce a population-based weighted graph model to assign weights to both nodes and links of a given unweighted physical level graph.

For a given graph $G = (N, L)$ with set of nodes N and links L, we introduce a weighting function $w(l)$ to assign weight to a given link $l \in L$. In physical-level graphs, nodes represent cities and links represent fiber cables. Our weight function utilizes city population and link betweenness to estimate link capacity. Although link capacity is measured in b/s, our objective is not to accurately determine the capacity but rather determine a correlated capacity function. We assume that nodes with higher population are more important than nodes with lower population. We also assume that a link between two nodes with high population should have higher capacity than a link between two nodes with low population unless they are transitions between other high capacity cities. The weight function $w(l)$ is calculated as:

$$w(l) = \text{Pop}(n_1) \times \text{Pop}(n_2) \tag{3.2}$$

where Pop(n) return the population of node n and the link l is connected by nodes n_1 and n_2.

To illustrate this model by a simple example, we present an 8-node physical-level network shown in Figure 3.2. For simplifications, we assign arbitrary small population values for each node while trying to keep them correlated to the actual population values as shown in Table 3.4. Observe that Kansas City, MO and St. Louis, MO have the highest population values. First, we calculate the population product and normalized betweenness as shown in Table 3.5. Using Equation 3.2, we calculate the assigned link weight based on the population product and link betweenness.

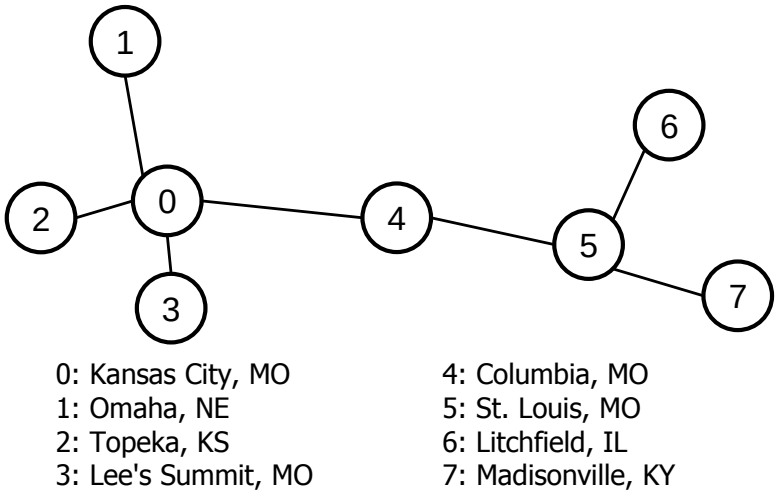

Figure 3.2: An 8-node physical-level network

As shown in Table 3.5, the highest assigned link weight is 5.71, which belongs to the link connecting Kansas City, MO and Columbia, MO. The second highest assigned link weight is 5.36, which belongs to the link connecting Columbia, MO and St Louis, MO. These two links have the highest weights because they carry a larger number of connections than other links as they have the highest betweenness value. We observe that these two links have higher assigned weights than the assigned weight for the link connecting Kansas City, MO and Omaha, NE, even though the latter has higher population product.

Table 3.4: Population-based weighted example cities and their population

Index	City	Population
0	Kansas City, MO, USA	5
1	Omaha, NE, USA	3
2	Topeka, KS, USA	2
3	Lee's Summit, MO, USA	2
4	Columbia, MO, USA	2
5	St. Louis, MO, USA	5
6	Litchfield, IL, USA	2
7	Madisonville, KY, USA	3

Table 3.5: Physical graph properties of three service provider networks

Link	Population Product	Normalized Betweenness	Assigned Weight
(0, 1)	15	0.25	3.75
(0, 2)	10	0.25	2.50
(0, 3)	10	0.25	2.50
(0, 4)	10	0.57	5.71
(4, 5)	10	0.54	5.36
(5, 6)	10	0.25	2.50
(5, 7)	15	0.25	3.75

3.1.7 Real-World Graphs Comparison

In this section, we compare our population-based model. We use two correlation functions to study the relationship between the real-world capacity values and our model-generated values. For linear comparison, we use the Pearson product-moment correlation coefficient, which yields 1 for perfect linear correlation, -1 for perfect inverse linear correlation, and 0 for no correlation [105]. For non-linear comparison, we use Spearman's rank correlation non-linear coefficient, which yields 1 for perfect correlation, -1 for perfect inverse non-linear correlation, and 0 for no correlation [106]. Here, we study the correlation between the actual capacity values of a real-world network and three generated values: link population Pop, link betweenness C_B, and population-betweenness

product Pop $\times$ C_B. The link population Pop is the product of the population of the two connecting nodes. The link betweenness C_B is the normalized betweenness value of the given link. the population-betweenness product Pop $\times$ C_B is the product of the link population and normalized betweenness. The correlation values are shown in Table 3.6, which shows inconsistent results. For KAREN and InternetMCI, the linear correlation for betweenness C_B indicate a higher dependency than both Pop and Pop $\times$ C_B. On the other hand, for CARNet, the linear correlation value is 0.79, which indicates that the actual link capacity values have high dependencies on link population. For GÉANT, the linear correlation values are relatively low for all generated values. For non-linear correlation, we see that actual link capacity values have high dependencies on link population. For InternetMCI, the non-linear correlation for betweenness C_B indicates higher dependency than the link population Pop. For CARNet, the non-linear correlation values show that population-betweenness product Pop $\times$ C_B has slightly higher dependencies than link population Pop and link betweenness C_B.

From some of these results, we observe that the generated capacity values are comparable to the actual capacity values. However, we speculate that these results are not consistent because of two reasons. First, our weighted real-world graph dataset has only educational networks, which are not necessarily designed to meet the demand of their city populations. Second, these networks are connected to other networks through one or multiple nodes, which makes the link betweenness values incorrect. For example, if CARNet network is connected to GÉANT thorough via multiple nodes, the link betweenness for a particular link for the combined network will be different compared to the two isolated graphs betweenness values.

Table 3.6: Correlation with real networks

Graph	Linear correlation			Non-Linear correlation		
	Pop	C_B	Pop $\times$ C_B	Pop	C_B	Pop $\times$ C_B
Karen	-0.19	0.24	-0.15	0.51	0.38	0.50
InternetMCI	-0.17	0.33	-0.15	0.00	0.43	0.03
CARNet	0.79	0.04	0.59	0.62	0.61	0.63
GÉANT	0.20	0.01	0.18	0.37	0.06	0.36

3.2 Centrality-Balanced Robustness

The high centrality nodes and links attract adversaries who can apply successful attacks on a targeted network by disrupting a few nodes with high centrality. For example, a star network has one central node with high degree. Attacking this node completely disconnects the communication network. To measure the graph balanced-centrality property, four metrics have been introduced [107]. First, the degree-balanced graph metric $\sigma^2_{C_D}$, is computed as the degree variance of all the nodes. Second, closeness-balanced graph metric $\sigma^2_{C_C}$, which is computed as the closeness variance of all the nodes. Third, node-betweenness-balanced graph metric $\sigma^2_{C_{B_v}}$, which is computed as node-betweenness variance of all the nodes. Fourth, link-betweenness-balanced graph metric $\sigma^2_{C_{B_l}}$, which is computed as node-betweenness variance of all the links.

3.3 Weighted Flow Robustness

The flow robustness metric, discussed in Section 2.2.1, considers only unweighted graphs. The weights in communication networks can represent several properties. For example, in physical level networks, they can represent fiber capacity while in wireless networks the can represent link reliability or bandwidth. Since communication networks can be modeled more realistically using weighted graphs, we propose a weighted flow robustness

graph metric that captures both connectivity and total link capacity in a given capacity-weighted graph. We assume that links with higher weights should contribute more to the value of flow robustness than links with lower weights. For example in physical level graphs, a fiber cut between two big cities with high link capacity should decrease the value of flow robustness more than a fiber cut between two cities with small link capacity. Before introducing the weighted flow robustness metric, we introduce a metric called adjusted total capacity (ATC) to measure the total capacity of all links. The ATC metric is defined as:

$$\mathrm{ATC}(G) = \sum w(l), \qquad l \in L \tag{3.3}$$

The weighted flow robustness (WFR) is defined as the product of the unweighted flow robustness (FR) and adjusted total capacity (ATC), which can be computed using:

$$\mathrm{WFR}(G) = \mathrm{FR}(G) \times \mathrm{ATC}(G) \tag{3.4}$$

Using several graphs, presented in Figure 3.3, we show how the weighted flow robustness (WFR) is computed using Equation 3.4. The results of calculating WFR are shown in Table 3.7.

Table 3.7: Calculating weighed flow robustness values

Graph	Weighed flow robustness(WFR)
G1	19.00
G2	11.00
G3	3.00
G4	3.33
G5	0.33
G6	0.00

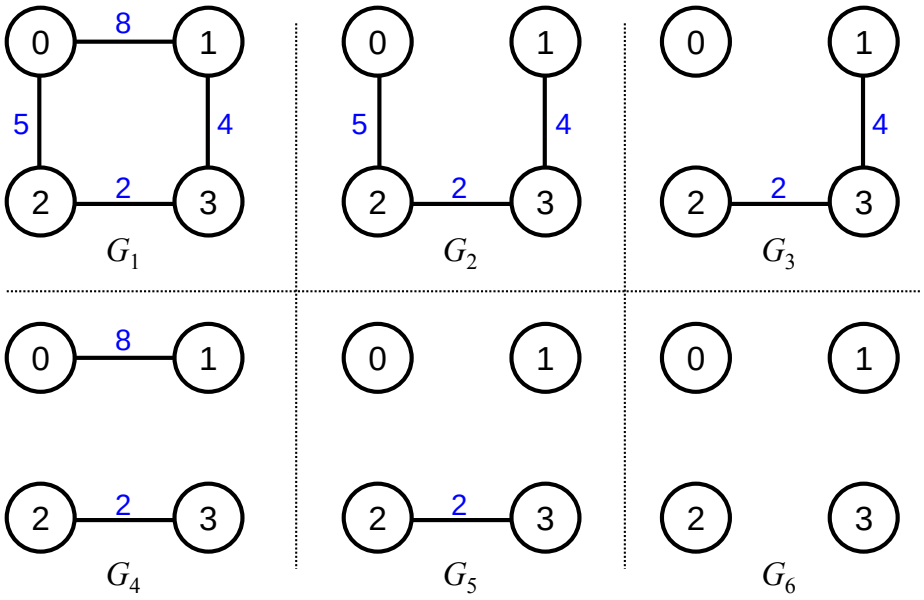

Figure 3.3: Weighed flow robustness calculation example

3.4 Graph Attack Models

We use a graph-theoretic model to attack a given graph and show how its flow robustness changes after each node removal. In this research, we use three centrality metrics: node betweenness, node closeness, and node degree [56]. Hence, we have three attack models, in which the node with the highest centrality is removed. The node-betweenness attack targets the node through which the highest number of shortest paths pass. The node-closeness attack targets the most central node in terms of hop count. The node-degree attack targets the node with the highest number of connections. If the attack requires removing multiple nodes, centrality metrics are recomputed upon attacking each node.

3.5 Measuring Network Resilience

In this section, we explain how to measure network resilience against a specific centrality-based attack. Flow robustness measures the reachability of a given graph. However, it

is not useful to distinguish between *connected* graphs. For example, the flow robustness value for a full-mesh graph and a star graph is identical, which in this case is one. As a solution, we introduce three robustness behavioral measures to calculate the sums of flow robustness of a given network resilience against centrality-based attacks. The robustness measures are: sums of flow robustness against degree attack (SFRD), sums of flow robustness against closeness attack (SFRC), and sums of flow robustness against betweenness attack (SFRB). Each measure captures resilience of a given network against the associated attack. For example, SFRD measures the network resilience against node degree-based attack.

Using an example, we illustrate how to measure network resilience of a 9-node wheel topology via sums of flow robustness against betweenness attack (SFRB). To compute the sums of flow robustness, we need to remove all nodes in this graph iteratively. In each iteration, one node is removed and the flow robustness is computed and added to the previous sum of flow robustness values. The permutation of the nodes list defines all possible ways for node attacks. In this example, we attack nodes based on their highest betweenness values, which yields the list {0, 1, 5, 3, 7, 8, 2, 4, 6}. Figure 3.4 depicts the topology while the attack is undergoing. Light green colored nodes indicate *connected* status (not attacked) while dark red nodes indicate *disconnected* status (attacked). Once a node is attacked all links attached to that node are removed. The values of robustness for reach iteration are shown in the Table 3.8. In step 2, we observe that after removing node 0, all 8 links are removed but flow robustness decreased by 0.22, which is not significant since there are alternative paths for the other nodes to communicate. However, in step 4, the flow robustness is decreased by $0.58 - 0.17 = 0.41$, which is the largest flow robustness decrease because the graph is partitioned into two components. Notice that we stop after step 6 since there are no remaining links. The sum of flow robustness values for a 9-node wheel topology is 2.61 as shown in the Table 3.8.

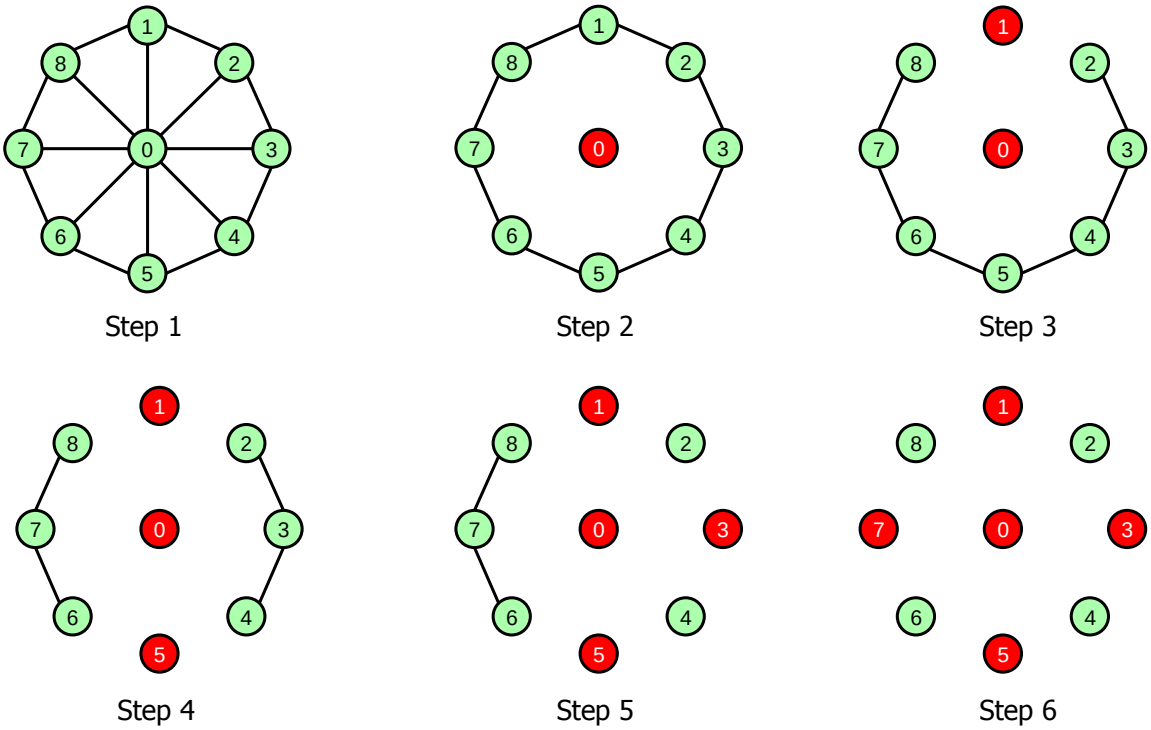

Figure 3.4: Measuring SFRB a 9-node wheel topology

Table 3.8: Measuring SFRB of a 9-node wheel topology

Step	Removed Nodes	FR	SFRB
1	{}	1.00	1.00
2	{0}	0.78	1.78
3	{0, 1}	0.58	2.36
4	{0, 1, 5}	0.17	2.53
5	{0, 1, 5, 3}	0.08	2.61
6	{0, 1, 5, 3, 7}	0.00	2.61

3.6 Summary

In this chapter, we presented our dataset, which includes baseline graphs, random graphs, and real-world graphs. Then, we introduced our graph robustness metrics and how they capture balanced-centrality of a given graph. Next, we presented our weighted flow ro-

bustness and showed how it is used to capture network resilience. We also introduced several centrality-based attack models and how they are used to measure network resilience. In addition, we discussed how our model converts unweighted graphs to weighted graphs using node populations and link betweenness. Finally, we present how we use flow robustness to measure network resilience against centrality attacks.

Chapter 4

Network Design and Improvement

In this chapter, we present our algorithms to improve a given graph based on a given objective function. We present four algorithms to improve algebraic connectivity, path diversity, balancing centrality, and a comprehensive comparison algorithm. The algebraic connectivity improvement algorithm is presented in Section 4.1. The work on improving algebraic connectivity has resulted in a publication [67]. The path diversity improvement algorithm is presented in Section 4.2, which has also resulted in a publication [108]. The balancing centrality improvement algorithm is introduced in 4.3,which has resulted in a publication [107]. The comprehensive comparison algorithm is presented in Section 4.4.

4.1 Algebraic Connectivity Improvement

Based on the greedy approach presented in Section 2.1.3, we develop an algorithm that improves the connectivity of a graph in terms of the algebraic connectivity metric by adding a set of links. Algebraic connectivity $a(G)$ is defined and discussed in Section 2.2.3. The main objective of our algorithm is to select the links that improve the algebraic connectivity of the graph in the least costly manner. Moreover, we introduce a cost-effect

parameter γ to our improvement algorithm to control the effect of algebraic connectivity and cost while selecting new links.

The greedy algorithm to increase algebraic connectivity in a graph is based on adding links to the nodes that have the least incident links (i.e. minimal degree nodes) [33, 35]. Our algorithm provides cost-efficient new links to improve network resilience measured by the algebraic connectivity metric. The assumptions, objective functions, and our heuristic algorithm is presented in Section 4.1.1. The evaluation of our algorithm on a sample graph is presented in Section 4.1.2.

4.1.1 Algebraic Connectivity Improvement Algorithm

In this section, we describe our algorithm that improves the algebraic connectivity and cost of a topology. Furthermore, we assume that node geolocations are given for a particular graph to which the improvement algorithm is applied, as would be the case for a deployed service provider.

Algorithm

The greedy improvement algorithm has three inputs: an input graph G_i, a number of required links L, and a cost-effect parameter γ. The input graph G_i has a number of nodes n_i with a number of links l_i. The number of required links L is the number of links that will be added to the graph. The cost-effect parameter γ is a tuning parameter between cost and algebraic connectivity. When $\gamma = 0$, the cost term of the rank function, shown in Equation 4.1, is neglected since it is zero. As a result, the algorithm selects the link that maximizes the algebraic connectivity. On the other hand, when $\gamma = 1$, the algebraic connectivity is neglected and the least link cost is selected in each iteration. The algorithm adds links to the graph with L iterations. To keep track of the selected

links in each iteration, the algorithm adds these links to a list. In each iteration, the algorithm starts by adding the selected links from previous iterations to the graph. Then, the rank value is computed for each candidate link and the link with the maximum rank value to be added is selected. A ranking function is used to select the best candidate in each iteration. The rank value r is computed using:

$$r = (1 - \gamma)a(G) + \gamma(1 - C) \tag{4.1}$$

where C represents the length of the ranked link.

The pseudocode of our algorithm is shown in Algorithm 1. This algorithm uses four functions: cost function $\mathsf{cost}(L)$, algebraic connectivity function $\mathsf{algConn}(G)$, link ranking function $\mathsf{maxLink}(D)$, and candidate link function $\mathsf{candidate}(G)$. The cost function $\mathsf{cost}(L)$ returns the cost of adding a link L. In this research, the cost is defined as the Euclidean distance between the two ends of the link. The algebraic connectivity function $\mathsf{algConn}(G)$ takes a graph G and returns the second smallest eigenvalue of its Laplacian matrix. The $\mathsf{maxLink}(D)$ function returns the maximum ranked link. The $\mathsf{candidate}(G)$ takes a graph G as input and returns a set of candidate links to be added to the graph. The candidate links are a set of links that are examined every time a link is added to a graph. One option to use for the candidate links is the set of complement links of a graph denoted as $\bar{G}$, which can be determined as the set of links in full mesh subtracted from the current links in a graph G. The number of complement links is computed as:

$$\frac{n_i(n_i - 1)}{2} - l_i \tag{4.2}$$

However, this number is computationally expensive as the number of nodes n_i gets larger, which results in a complexity of $O(Ln_i^2)$. In an attempt to decrease the number of candidate links, we only examine the links connected to the lowest degree node in the graph. As a result, the algorithm complexity decreases to $O(Ln_i)$.

Functions:
cost(L) := cost function
algConn(G) := algebraic connectivity function
candidate(G) := candidate links function
maxLink(D) := max value of a dictionary
Input:
G_i := input graph
L := number of required links
γ := cost effect parameter
Output:
an ordered list of the added links

```
begin
    selectedLinks = []; empty ordered list
    rank = {}; empty dictionary
    while L > 0 do
        G = G_i
        G.addlinks(AddedLinks)
        for link in candidate(G) do
            rank[link]=(1 − γ)algConn(G) + γ(1−cost(link))
        end
        selectedLinks.add(maxLink(rank))
        L = L − 1
    end
    return selectedLinks
end
```

Algorithm 1: Algebraic connectivity Improvement algorithm

Both the algConn(G) and cost(L) functions are normalized to have a maximum value of one. Since the theoretical maximum value for the algebraic connectivity of a given graph is the number of its nodes, it is normalized by dividing it by the number of nodes. To normalize the cost function, it is divided by the maximum possible distance between any nodes in the graph.

4.1.2 $a(G)$ Improvement Algorithm Example

In this section, we explain how our greedy algorithm improves a topology on a small-size graph. Figure 4.1 shows a sample graph with 8 nodes and 9 links as solid lines. The initial algebraic connectivity of this sample graph is 0.3432 and the initial cost (i.e. total link length in km) of the graph is 8,203. Our greedy algorithm adds links to the least connected nodes, which in the example are nodes 0 and 7. The six candidate links for node 0 are shown as square dots, whereas five candidate links for node 7 are shown as long dashes and dots. Throughout this example, we describe how our algorithm operates if we are going to add *one* link $L = 1$ to the sample graph shown in Figure 4.1.

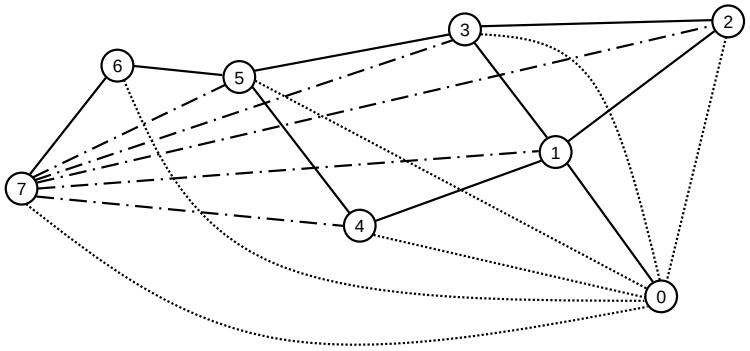

Figure 4.1: Graph example for algebraic connectivity based improvement

There can be a maximum of 28 links in this 8-node graph (the maximum number of links can be calculated by $\frac{n(n-1)}{2}$). Since there are 9 links in the graph, if we were to examine all links in the complement set, there would be $28 - 9 = 19$ candidate links. In

the sample graph shown in Figure 4.1, there are six candidate links that can be added to node 0 and there are five links for node 7 using our greedy algorithm. Therefore, the candidate link set is reduced to 11, because our algorithm only considers candidate links from the least connected nodes. The algebraic connectivity and cost value of adding each link individually for $\gamma = 0$ and $\gamma = 1$ is shown in Table 4.1.

Table 4.1: $a(G)$ and cost values for the example graph

Link	$\boldsymbol{\gamma = 0}$		$\boldsymbol{\gamma = 1}$	
	$\boldsymbol{a(G)}$	$\boldsymbol{\Delta a(G)}$	**cost**	**Δ cost**
$0 \leftrightarrow 2$	0.3485	0.0053	**9,275**	**1,072**
$0 \leftrightarrow 3$	0.3588	0.0156	9,405	1,202
$0 \leftrightarrow 4$	0.3659	0.0227	9,848	1,645
$0 \leftrightarrow 5$	0.4079	0.0647	10,624	2,421
$0 \leftrightarrow 6$	0.5908	0.2476	11,228	3,025
$0 \leftrightarrow 7$	0.7713	0.4281	11,843	3,640
$7 \leftrightarrow 1$	**0.8345**	**0.4913**	11,302	3,099
$7 \leftrightarrow 2$	0.7071	0.3639	12,061	3,858
$7 \leftrightarrow 3$	0.6651	0.3219	10,915	2,712
$7 \leftrightarrow 4$	0.5918	0.2486	10,207	2,004
$7 \leftrightarrow 5$	0.5075	0.1643	9,463	1,260

When $\gamma = 0$, our algorithm ignores the cost associated with adding a link and selects the additional link that increases the algebraic connectivity of the graph the most. For $\gamma = 0$, the algorithm adds the link between node 1 and 7 in the example graph since it provides the highest algebraic connectivity among the 11 candidate links. When $\gamma = 1$, the cost is the dominant factor determining the addition of a link. Therefore, our heuristic algorithm selects the link between node 0 and 2, since it incurs the lowest cost among the candidate set of links. The selection of links via our heuristic algorithm is highlighted in bold in Table 4.1.

4.2 Path Diversity Improvement

In this section, we first develop an algorithm to calculate the TGD (described in Section 2.2.4) of a graph [11,32]. We modify the TGD calculation algorithm so that instead of considering relatively more diverse paths [11,32], we consider the effect of the diversity of all paths when calculating the TGD. Second, we introduce an algorithm for finding the optimal k-diverse paths considering both nodes and links using an exhaustive path search. Lastly, we present an improvement algorithm that improves the path diversity of a graph based on the TGD metric. Our graph improvement algorithm considers adding links with the least cost among available choices.

The rest of the section is organized as follows: The algorithm to calculate the path diversity of a graph is explained in Section 4.2.1. The assumptions, objective functions, and our heuristic algorithm is presented in Section 4.2.2. The evaluation of our algorithm on a sample graph is presented in Section 4.2.3.

4.2.1 Finding k-Diverse Paths

In this section, we present a new k-diverse paths algorithm that determines the k paths between a source s and destination d. This algorithm has four inputs: a source node s, a destination node d, a hop-count threshold h, and a threshold for the number of returned diverse paths k. Moreover, this algorithm uses four functions: all_simple_paths(s,d,h), sort(L), path2elements(P), and p_div(P). The all_simple_paths(s,d,h) function finds all possible loopless paths between source s and destination d, with hop-count threshold h for the path length. If h is not set, all possible paths are returned. Whereas selecting higher h value yields more accurate results, it requires more computing resources. The number of all possible simple paths can be as large as $n!$, where n is the number of nodes. This number is infeasible to compute for large size graphs, thus, the h parameter should

be chosen based on the size of the graph and available computing resources. The sort(L) function sorts a list of tuples of three elements: link, diversity, and cost. The sorting is done in decreasing order of the diversity value and increasing order of the cost value for links with equal diversities. The path2elements(P) function converts a path P to a set of nodes and links elements. The p_div(P) function computes the diversity of the path with respect to the selected_elements set. The pseudo code is shown in Algorithm 2.

Functions:
all_simple_paths(s,d,h) := all simple paths between node s and node d with a threshold hop-count h
sort(l) := sorting l function
path2elements(P) := path P to link and node elements
p_div(P) := computes path diversity of path P
Input:
s := source node
d := destination node
h := hop-count threshold value for examined paths
k := threshold value for returned diverse paths
Output:
an ordered list of $k$diverse paths
begin
    diverse_paths = []; empty ordered list
    selected_elements = {}; empty set
    for path in all_simple_paths(s,d,h) **do**
        diverse_paths.append(path,p_div(path),len(path))
        selected_elements.add(path2elements(path))
    end
    sort(diverse_paths)
    return diverse_paths[0:k]
end

Algorithm 2: k-diverse path algorithm

This algorithm has two phases: finding all simple paths and finding the most diverse k paths. In the first phase, all possible paths are determined between a source s and destination d with a hop-count threshold h using the function all_simple_paths(s,d,h). For the second phase, the algorithm determines the most diverse paths among the returned paths via the all_simple_paths function. The shortest path P_0 is added to the selected paths

in the first iteration and its elements (nodes and links) are added to the selected_elements set. Next, the algorithm iterates over the remaining paths by computing the diversity of the path using the p_div(P) function and adding it along with the path length to the diverse_paths list while the path elements are added to the selected_elements. Finally, using the sort(L) function, all the tuples in the diverse_paths list are sorted in decreasing order of their diversity. In case there are multiple paths with the same diversity, these paths are sorted in increasing order of their hop-count.

4.2.2 Path Diversity Improvement Algorithm

In this section, we describe our algorithm that improves the TGD of a given graph with its node locations by adding new cost-efficient links. Our heuristic algorithm is implemented using Python and uses the NetworkX library [109] for graph algorithms. The objective of this algorithm is to improve the TGD of a graph by adding a user-specified number of links. The algorithm adds one link that increases the TGD the most. If there are multiple links that give the same largest TGD value, the least costly link is selected. We measure the cost of a link in terms of the Euclidian distance of that link. The link addition process is repeated until the number of links requested by the user is added.

Algorithm

The objective of this algorithm is to improve the TGD of a graph by adding a user-specified number of links. The algorithm adds one link that maximizes the TGD value. If there are multiple links that give the same largest TGD value, the least costly link is selected. We measure the cost of a link in terms of the Euclidean distance of that link.

The topology improvement algorithm has two inputs: an input graph G_i, a number of required links L_r. The input graph G_i has a number of nodes n_i with a number of links

l_i. The number of required links L_r is the number of links that should be added to the graph. The algorithm adds links to the graph with L_r iterations. To keep track of the selected links in each iteration, the algorithm adds this link to the selectedLinks list. In each iteration, the algorithm starts by adding the selected links from previous iterations to the graph.

The candidate set contains the links that are connected to the pairs with the lowest EPD values of the graph and not currently present in the graph. To find the best candidate link, each link in the candidate set is added to the graph and the EPD of the corresponding pair is computed and mapped to that link. Then, the link with the largest EPD is selected. In case there are multiple links with the same largest EPD, the least costly link is selected. This process is repeated until the user requested number of links are added.

This algorithm uses four functions: cost(l), epd(P), candidate(G), and bestLink(L). The cost function cost(l) returns the cost of adding a link l. In this research, the cost is defined as the Euclidean distance between the two ends of the link. The effective path diversity function epd(P) computes the effective path diversity of the path P based on Equation 2.22. The bestLink(L) function returns the link with the highest EPD and lowest cost in case of multiple highest EPD values. The candidate(G) takes a graph G as input and returns a set of candidate tuples of two elements. The first element is a lowest EPD pair and the second element is a candidate link. The candidate links are the set of all links in which one end is connected to a node in the lowest EPD pair and the other end is connected to a node in the graph given that this link does not exist in the graph. For each pair and link in the candidate set, we add the link to the graph and compute the new EPD value of that pair with its cost. Finally, the link with the highest EPD and the lowest cost is selected using bestLink(L) function and then added to the selectedLinks list. The pseudo code is shown in Algorithm 3.

Functions:
cost(l) := cost of link l
epd(P) := EPD value of path P
candidate(G) := candidate links function
bestLink(L) := maximum EPD value of links list L
Input:
G_i := input graph
L_r := number of required links
Output:
an ordered list of the added links

```
begin
    selectedLinks = []; empty ordered list
    links_epd_list = []; empty ordered list
    while L_r > 0 do
        G = G_i
        G.addlinks(selectedLinks)
        for P, L  in candidate(G) do
            links_epd_list.append((L,epd(P),cost(L)))
        end
        selectedLinks.add(bestLink(links_epd_list))
        L_r = L_r − 1
    end
    return selectedLinks
end
```

Algorithm 3: Topology improvement algorithm

4.2.3 Path Diversity Improvement Algorithm Example

In this section, we explain how our heuristic algorithm improves a topology on a small-size graph. Figure 4.2 shows a sample graph with 5 nodes and 5 links. In this example, the hop count threshold h is set to 10 and the number of diverse paths k is set to 4. The initial TGD value of this sample graph is 0.2023. Our heuristic algorithm examines the links connected to the least EPD pairs. The smallest EPD pairs are (1,2) and (3,4) with EPD of 0 since they have no alternative paths. Therefore, the candidate set consists of four possible links for each pair.

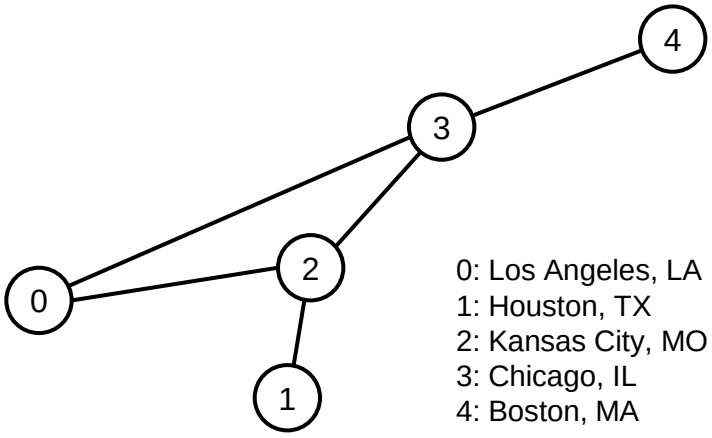

Figure 4.2: Graph example for path diversity based improvement

To find the best candidate, we determine the resulting EPD of the corresponding pair after adding the candidate link and the cost incurred as shown in Table 4.2. Then, we find the link that gives the highest pair EPD. Among the eight candidates, there are four links that give a high pair EPD of 0.50. The next step is to find the lowest length link, which is the link (1,3). After adding this link, the new TGD of this graph is 0.4034, which is almost double the initial TGD.

Table 4.2: EPD and cost values for the candidate links in the example graph

Node Pair	Link	Pair EPD	Cost
(1, 2)	(1, 0)	0.50	2,177
(1, 2)	(1, 3)	**0.50**	**1,043**
(1, 2)	(1, 4)	0.46	2,311
(1, 2)	(2, 4)	0.00	1,988
(3, 4)	(3, 1)	0.00	1,043
(3, 4)	(4, 0)	0.50	4,058
(3, 4)	(4, 1)	0.46	2,311
(3, 4)	(4, 2)	0.50	1,988

4.3 Balancing Centrality Improvement

In this section, we describe our algorithm that balance the centrality of a given graph based on a given centrality function. The centrality functions used in this research are node betweenness, node closeness, and node degree. The objective of this algorithm is to balance the centrality by minimizing the centrality variance across all the nodes of a given graph via adding a number of links constrained by a cost budget. To achieve that, our algorithm minimizes the variance of the nodes' centralities measured by one of three node centrality functions: node betweenness, node closeness, and node degree. If there are multiple links that yield the same minimum variance value, the lowest cost link is selected. Moreover, if there are multiple links with the same cost, the link with a lower index is selected. We measure the cost of a link in terms of the Euclidean length of that link. The link addition process is repeated until one of two conditions is met. First, if there is no link that can be added without exceeding the budget, the algorithm stops because the budget constraint is met at this point. Second, whenever all candidate links are added, the algorithm stops because there are no more links to be added to the graph.

4.3.1 Balancing Centrality Improvement Algorithm

The objective of this algorithm is to balance graph centrality among all the nodes of a given graph by adding a set of links constrained by a cost budget. To achieve this objective, our algorithm minimizes the variance of the node centralities measured by one of the three node centrality functions: node betweenness, node closeness, and node degree. If there are multiple links that yield the same minimum variance value, the lowest cost link is selected. The pseudocode of our algorithm is shown in Algorithm 4.

There are two inputs for this algorithm: an input graph G and a budget constraint B. The input graph G has a number of nodes n with a number of links l and the node positions. The budget constraint B is measured in meters to specify the allowed total length for link addition. The algorithm adds links to the graph iteratively. To keep track of the selected links in each iteration, the algorithm adds these links to the selectedLinks list. Moreover, to keep track of the selected links cost, the algorithm increments the totalCost by the cost of each added link. For the candidate set, all possible candidate links are in the graph's complement. However, this set may contain very long links that are not practical to be added to a physical graph. Thus, the candidate links are constrained by the longest link in the input graph as discussed in Section 3.1.5.

There are seven functions used by this algorithm. The cost function cost(l) returns the cost of adding a link l that is defined as the Euclidean distance between the two ends of the link. The function nBtwn(G) computes the node betweenness of all the nodes in graph G. The function nClos(G) computes the closeness for every node in graph G. The function nDeg(G) computes the number of links connected to every node in graph G. The bestLink(L) function returns the best candidate link given that it is an affordable link and it has the minimum-variance value and lowest cost in case of multiple tie minimum-variance values. The var(L) function returns the variance of the values in list L. The

Functions:
cost(l) := cost of link l
nBtwn(G) := betweenness for all nodes in graph G
nClos(G) := closeness for all nodes in graph G
nDeg(G) := degree for all nodes in graph G
candidate(G) := candidate links function
var(L) := computes variance of list L
bestLink(L) := affordable low variance in list L
Input:
G := input graph
B := available budget
Output:
selectedLinks := an ordered list of the selected links
begin
    centralityFunc = nBtwn | nClos | nDeg
    selectedLinks = [] ; empty ordered list
    varAndCost = [] ; empty ordered list
    totalCost = 0; initial total cost is zero
    while B >= totalCost $\wedge$
    selectedLinks $\neq$ candidate(G) **do**
        G.addlinks(selectedLinks)
        for l in candidate(G) **do**
            centralityVar = var(centralityFunc(G)) varAndCost.append((l, centralityVar, cost(l)))
        end
        selectedLink = bestLink(varAndCost)
        selectedLinks.add(selectedLink)
        totalCost += cost(selectedLink)
    end
    return selectedLinks
end

Algorithm 4: Balancing centrality algorithm

candidate(G) function returns candidate links in graph G.

The centrality function centralityFunc is selected from the three options: nBtwn, nClos, and nDeg. For each link in the candidate set, the algorithm temporarily adds the link to the graph and computes the variance value and the cost incurred by that link, which are added to the varAndCost list. After that, the temporary link is removed and the next candidate link is added to undergo the same process. The link with the minimum variance and the lowest cost is selected using the bestLink function and then it is added to the selectedLinks list. Inside the bestLink function, if the minimum variance link cost plus the total cost exceeds the budget, a next minimum link is examined until a link with affordable cost is found. This process is repeated until no link can be added without exceeding the given budget or there are no more available links in the candidate set.

4.3.2 Balancing Centrality Algorithm Example

In this section, we explain how our heuristic algorithm improves a topology on a small-size graph. Figure 4.3 shows a sample graph connecting major U.S. cities with 7 nodes and 7 links. In this example, we apply our algorithm to add a single link using three objective functions one at a time. In this example, we add a maximum length constraint for the links located in the candidate set as discussed in Section 3.1.5.

The objective functions are minimizing the betweenness variance of the sample graph nodes, minimizing the closeness variance of the sample graph nodes, and minimizing the degree variance of the sample graph nodes. To find the best candidate, the algorithm finds the candidate set, which contains the links of the complement graph that are not longer than the current maximum link in the graph. The number of links in the complement graph is $\frac{7\times6}{2} - 7 = 14$ links. In this example, the longest link of the input graph is between nodes 5 and 6 that has a length of 2,177 km. Therefore, 8 links that have a

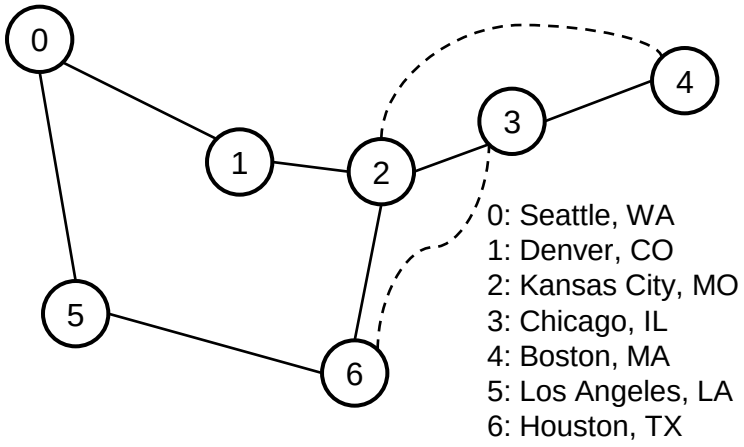

Figure 4.3: Improvement graph example

length greater than this value are removed from the candidate link set and 6 links remain in the candidate set. For each candidate link, the algorithm determines the centrality variance across all nodes after adding the candidate link and the cost incurred as shown in Table 4.3.

Table 4.3: Centrality variance and cost for candidate links

Candidate links	**Betweenness variance**	**Closeness variance**	**Degree variance**	**Cost [m]**
(1, 3)	0.0125	0.0085	0.4897	1,453,452
(1, 5)	0.0374	0.0086	0.4897	1,259,832
(1, 6)	0.0323	0.0099	0.4897	1,070,221
(2, 4)	0.0379	**0.0076**	0.4897	1,988,059
(2, 5)	0.0459	0.0115	0.7755	2,143,391
(3, 6)	**0.0125**	0.0085	**0.4897**	**1,043,873**

According to our algorithm, the link resulting in the minimum variance is selected. For example, using closeness variance as an objective function, the link (2,4) is selected because it gives the minimum variance value of 0.0076. In case there are multiple minimum values, the link with the least cost is selected among those minimum variance links. For example, using betweenness variance as an objective function, there are two links with

the minimum value of 0.0125, namely links (1,3) and (3,6). Next, the algorithm finds the link with the minimum cost, which is (3,6) in this example.

4.4 Comprehensive Comparison

In this section, we present a *generic* greedy algorithm to improve a given graph robustness by adding a set of links to maximize a given graph robustness metric. The objective of this algorithm is to improve a given graph robustness metric by adding a specific set of links while using a lower cost of links as a tie breaker. The output of this algorithm is an improved graph for a given robustness metric . By applying this algorithm to all robustness metrics with a constant number of links, we obtain improved graphs based on each robustness function. We study and compare the network resilience of the yielded improved graphs against targeted attacks.

4.4.1 Algorithm

We use a greedy approach to construct our algorithm, which adds a set of links L_i to a given graph G_i. The main objective of this algorithm is to determine a set of links of size L_i to maximize or minimize a given robustness metric. The algorithm adds the links iteratively by selecting the link that satisfies the objective function such as maximizing algebraic connectivity or minimizing the network criticality. The pseudocode of our algorithm is shown in Algorithm 5.

Our improvement algorithm uses four functions: $\mathsf{robustness}(G)$, $\mathsf{cost}(l)$, $\mathsf{candidate}(G)$, and $\mathsf{bestLink}(L)$. The $\mathsf{robustness}(G)$ function returns the robustness of a given graph G. This function is selected as one of the robustness functions such as algebraic connectivity and network criticality. The cost function $\mathsf{cost}(l)$ returns the cost of adding a link l that is defined as the Euclidean distance between the two ends of the link. The $\mathsf{candidate}(G)$

Functions:
robustness(G) := objective robustness function
cost(l):= cost of link l
candidate(G) := candidate links function
bestLink(L) := highest improvement link in list L
Input:
G_i := input graph
L_i := number of required links
Output:
selectedLinks := an ordered list of the selected links

```
begin
    selectedLinks = [] ; empty ordered list
    while L_i > selectedLinks.length() do
        G.addlinks(selectedLinks)
        iterationList = []
        for l in candidate(G) do
            improvement = robustness(G) iterationList.append((l,
            improvement,cost(l)))
        end
        selectedLink = bestLink(iterationList)
        selectedLinks.add(selectedLink)
    end
    return selectedLinks
end
```

Algorithm 5: Comprehensive comparison improvement algorithm

function returns a set of the candidate links, from which the subset of length L_i is selected. The bestLink(L) function returns the link with the highest improvement value while a low cost link is considered as a tie breaker to obtain cost-efficient results. For determining the candidate set using the candidate(G) function, all possible candidate links are located in the input graph complement constrained by the longest link as discussed in Section 3.1.5.

4.5 Summary

In this chapter, we presented four improvement algorithms to improve network resilience against targeted attacks. First, the algebraic connectivity improvement algorithm was presented to add links to improve maximize algebraic connectivity, which improves the resilience against graph partitioning. In addition, the path diversity improvement algorithm was presented to improve network path diversity in terms of increasing the number of disjoint paths. Third, the balancing centrality improvement algorithm was introduced to add links to balance a given graph centrality such as degree, closeness, and betweenness. Finally, the comprehensive comparison algorithm was presented to add links to a given graph via a specified robustness function.

Chapter 5

Network Resilience Evaluation

In this chapter, we present our evaluation of the graph robustness metrics discussed in Section 2.2. In Section 5.1, we start by studying the execution time for these graph robustness metrics as the number of nodes increases and provide the results. In Section 5.2, we present our evaluation results of applying our algebraic connectivity improvement algorithm. In Section 5.3, we present our evaluation results of applying our path diversity improvement algorithm. In Section 5.4 shows the evaluation results of our balancing centrality algorithm. In Section 5.5, we present our accuracy and improvement evaluation of the spectral robustness graph metrics. Moreover, we conduct a comprehensive comparison of all robustness graph-metrics to measure their accuracy in predicating network resilience against centrality-based attacks in Section 5.7. Finally, we study non- and improved graphs for all metrics and evaluate their network resilience against such attacks in Section 5.8.

5.1 Graph Metrics Time Complexity

In this section, we study the execution time for the graph robustness metrics presented in Section 2.2. Graph-robustness metric algorithmic complexity generally depends on

Table 5.1: Graph metrics execution time in seconds

	10	20	30	40	50	60	70	80	90	100
$\bar{C}_D$	2.57e-5	2.93e-5	3.46e-5	3.86e-5	4.29e-5	4.88e-5	5.10e-5	5.77e-5	6.47e-5	6.71e-5
$\sigma^2_{C_D}$	8.02e-5	8.39e-5	9.07e-5	9.52e-5	1.01e-4	1.08e-4	1.09e-4	1.16e-4	1.23e-4	1.27e-4
$\sigma^2_{C_C}$	2.65e-4	7.27e-4	1.50e-3	2.64e-3	4.09e-3	6.13e-3	8.27e-3	1.15e-2	1.55e-2	1.97e-2
$\sigma^2_{C_{B_v}}$	5.17e-4	1.82e-3	4.59e-3	8.85e-3	1.50e-2	2.32e-2	3.38e-2	4.74e-2	6.66e-2	8.74e-2
$\sigma^2_{C_{B_1}}$	5.89e-4	2.20e-3	5.60e-3	1.07e-2	1.82e-2	2.85e-2	4.16e-2	5.95e-2	8.32e-2	1.09e-1
CC	9.52e-5	3.15e-4	7.19e-4	1.39e-3	2.25e-3	3.83e-3	5.60e-3	8.53e-3	1.17e-2	1.52e-2
As	4.33e-4	6.86e-4	1.04e-3	1.46e-3	2.02e-3	2.64e-3	3.45e-3	4.29e-3	5.25e-3	6.27e-3
R	1.82e-4	6.31e-4	1.38e-3	2.52e-3	3.96e-3	6.01e-3	8.18e-3	1.14e-2	1.51e-2	1.96e-2
D	1.81e-4	6.31e-4	1.38e-3	2.51e-3	3.96e-3	6.01e-3	8.21e-3	1.15e-2	1.52e-2	1.97e-2
$\bar{d}$	1.89e-4	6.21e-4	1.37e-3	2.47e-3	3.86e-3	5.64e-3	7.97e-3	1.10e-2	1.48e-2	1.92e-2
TGD	1.85e-2	1.3200	9.5300	3.87e1	1.04e2	2.36e2	4.57e2	8.55e2	1.42e3	2.18e3
λ_2	1.09e-3	1.20e-3	1.37e-3	1.62e-3	1.92e-3	2.26e-3	2.72e-3	3.42e-3	4.69e-3	5.46e-3
$\Delta\lambda$	5.54e-4	8.15e-4	1.17e-3	1.63e-3	2.31e-3	3.11e-3	4.19e-3	5.49e-3	7.01e-3	8.78e-3
$\hat{\tau}$	1.22e-3	1.53e-3	1.97e-3	2.63e-3	3.55e-3	5.06e-3	7.42e-3	7.42e-3	9.43e-3	1.30e-2
WS	1.81e-3	1.94e-3	2.15e-3	2.43e-3	2.80e-3	3.19e-3	3.72e-3	4.47e-3	5.10e-3	6.49e-3
$\bar{\lambda}$	5.57e-4	8.17e-4	1.19e-3	1.67e-3	2.29e-3	3.13e-3	4.44e-3	5.55e-3	7.15e-3	8.90e-3
C^*	1.09e-3	1.22e-3	1.42e-3	1.67e-3	1.98e-3	2.36e-3	2.88e-3	3.56e-3	4.20e-3	4.89e-3

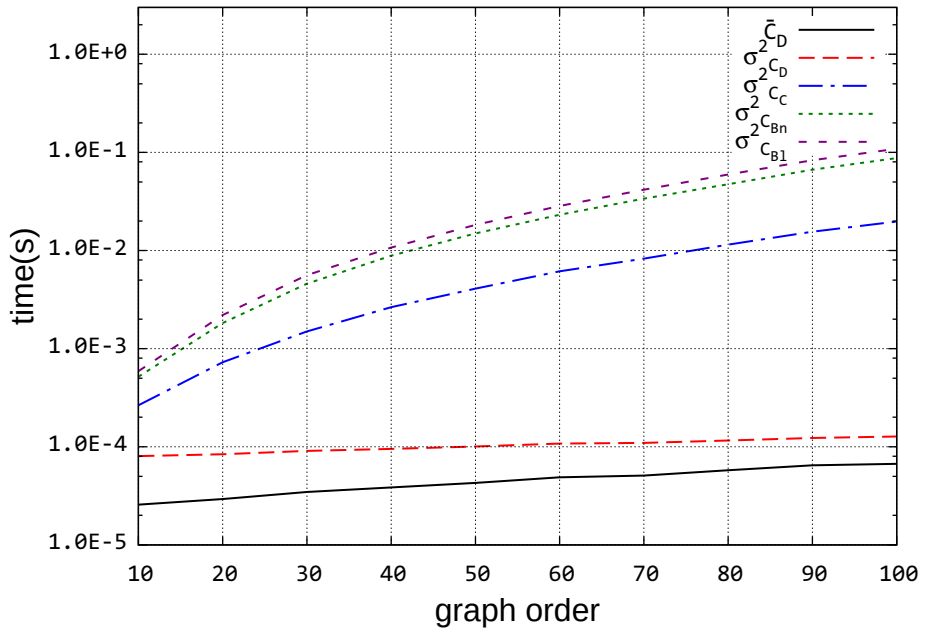

Figure 5.1: Execution time for computing centrality-balanced graph metrics

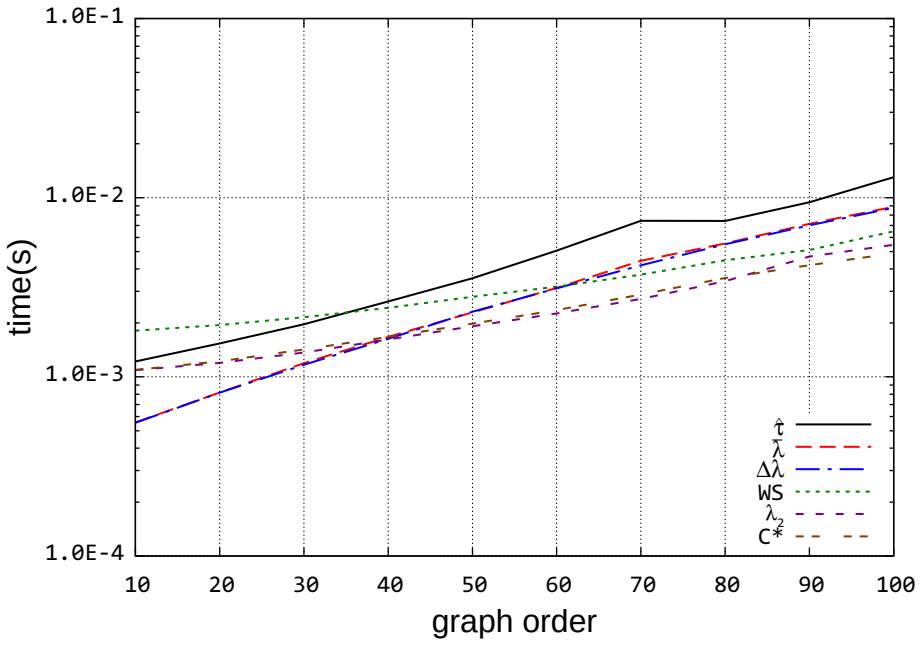

Figure 5.2: Execution time for computing spectral graph metrics

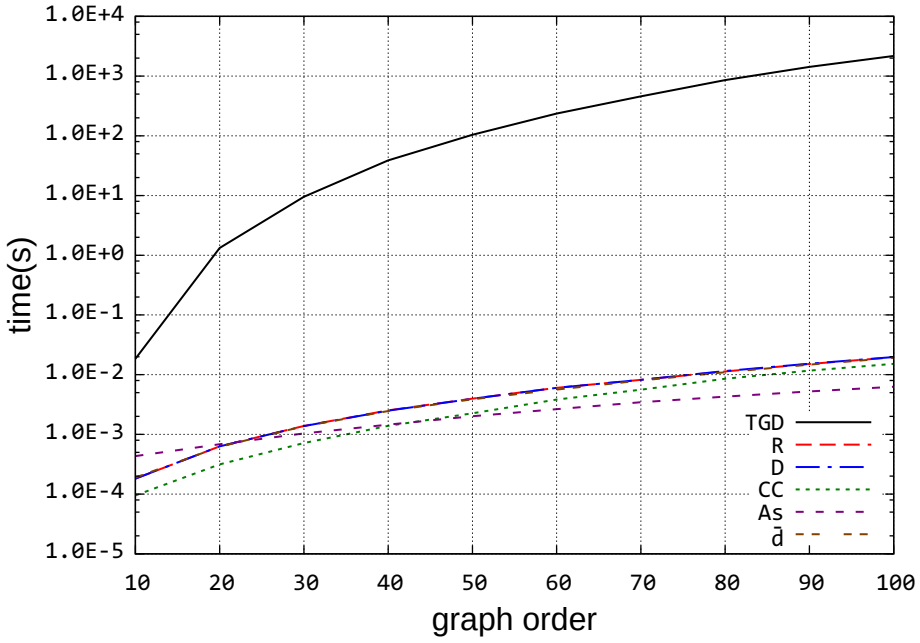

Figure 5.3: Execution time for computing uncategorized graph metrics

graph order (number of nodes) as discussed in Section 2.2. Several computer network design and improvement approaches require a low execution time, particularly when

dealing with dynamic network infrastructure. Hence, we study the execution time for each robustness metric as the number of nodes increases. All graph metric robustness function implementations are from the Python NetworkX library [109]. The number of nodes are between 10 and 100 with an increment of 10 nodes. The execution time is measured in seconds and the average results of 20 samples for all graph metrics are shown in Table 5.1. Moreover, we plot our results as a function of graph order to show how the execution time behaves as number of the nodes increases. To present our plots more clearly, we divide graph robustness metrics into three categories: centrality-balanced, spectral, and uncategorized.

The execution time for computing centrality-balanced graph metrics is shown in Figure 5.1. For these graph metrics, execution time takes from 10^{-5} to 1 second as shown in Figure 5.1. However, we observe that the variances of link and node betweenness requires more time than the average degree, the variances of node degree, and clustering coëficcient as graph order increases. The execution time for computing spectral graph metrics is shown in Figure 5.2. Among the three categories, spectral graph metrics show the fastest execution time with less than 200 ms as the graph order reaches 100 nodes. The execution time for computing uncategorized graph metrics is shown in Figure 5.3. The total graph diversity (TGD) is the most expensive graph robustness metric as it exceeds 1000 s as the as the graph order reaches 100 nodes.

5.2 Improvement Algebraic Connectivity

In this section, we apply our algebraic connectivity algorithm, presented in Section 4.1.1, on the dataset presented in Section 3.1 First, we study and analyze the impact of the algorithm on algebraic connectivity and cost. Then, we evaluate this algorithm by apply-

ing a set of centrality-based attacks on the non- and improved graphs to examine their resilience via flow robustness values.

5.2.1 Improvement Analysis

Our algorithm is applied to five service providers by adding 100 links. We show the graph algebraic connectivity and the cost incurred in terms of meters after adding each link. Moreover, we show the relation of cost and algebraic connectivity, and the slope in these figures indicates how the cost increases as the graph connectivity improves.

Selection of γ values

The value of γ parameter ranges 0 to 1 and controls the outcome of the algorithm as described in Section 4.1.1. In Equation 4.1, we have two terms: $(1-\gamma)a(G)$ and $\gamma(1-C)$. The $a(G)$ is a normalized algebraic connectivity value, which is low for sparse graphs and a value of one for a full mesh graph. The value of C denotes the normalized cost of adding a link and it is low when the maximum possible link length in the input graph is larger than the average link length in the candidate set. Therefore, choosing the value of γ depends on the initial properties of the input graph. For the five graphs presented in Table 3.1.3, we choose for $\gamma = \{0, 10^{-9}, 10^{-7}, 10^{-5}, 1\}$ because the cost term is larger than the γ term by about six degrees of magnitude.

Physical-level topology analysis

We have added a link length constraint to our heuristic algorithm to discard the links that are longer than the actual maximum link of the graph. Algebraic connectivity improvement of the Sprint physical level topology after adding 100 links iteratively is depicted in Figure 5.4. The algebraic connectivity is higher for $\gamma = 0$ than the other

values of γ, and for $\gamma = 1$ our algorithm considers minimizing the cost, but not improving the algebraic connectivity. We observe a possible phase transition occurs when $\gamma = 1$ for the physical-level graphs. For example, algebraic connectivity improvement of the Sprint physical topology starts with a moderate increase, and after the 20th link addition, the improvement (i.e. the slope of the curve) gets steeper. The reasons for the occurrence of this phenomenon will be the subject of future work.

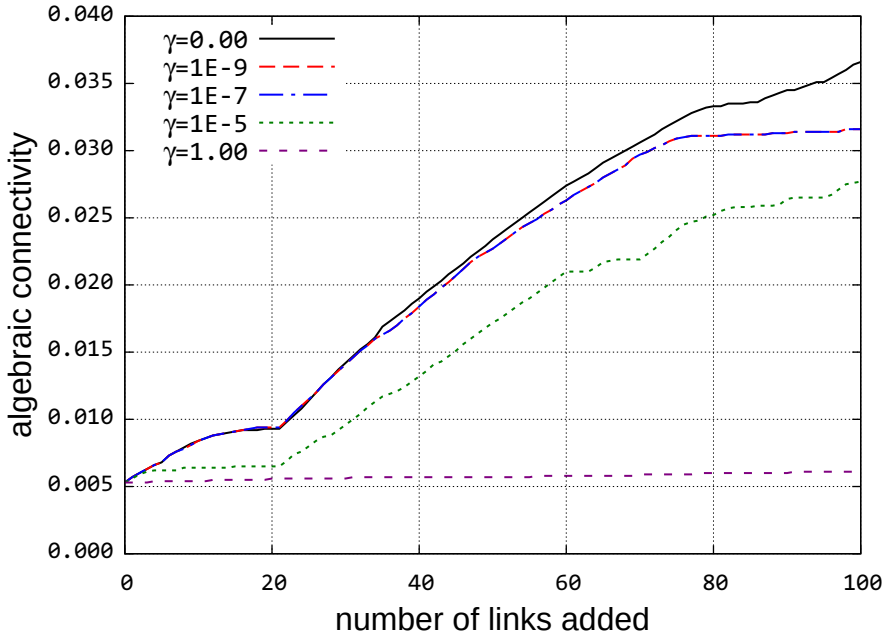

Figure 5.4: Connectivity improvement for Sprint physical topology

The cost incurred when adding 100 links iteratively to the Sprint physical level topology is shown in Figure 5.5. The cost in physical topology is the length of links to be laid between nodes, thus, short links are favorable in physical level topology improvement for $\gamma = 1$.

The relationship between connectivity and cost for the Sprint physical level topology is shown in Figure 5.6. For the Sprint example shown in Figure 5.6, if the cost is the constraint (i.e. $\gamma = 1$), the designer can improve the algebraic connectivity to 0.006 by

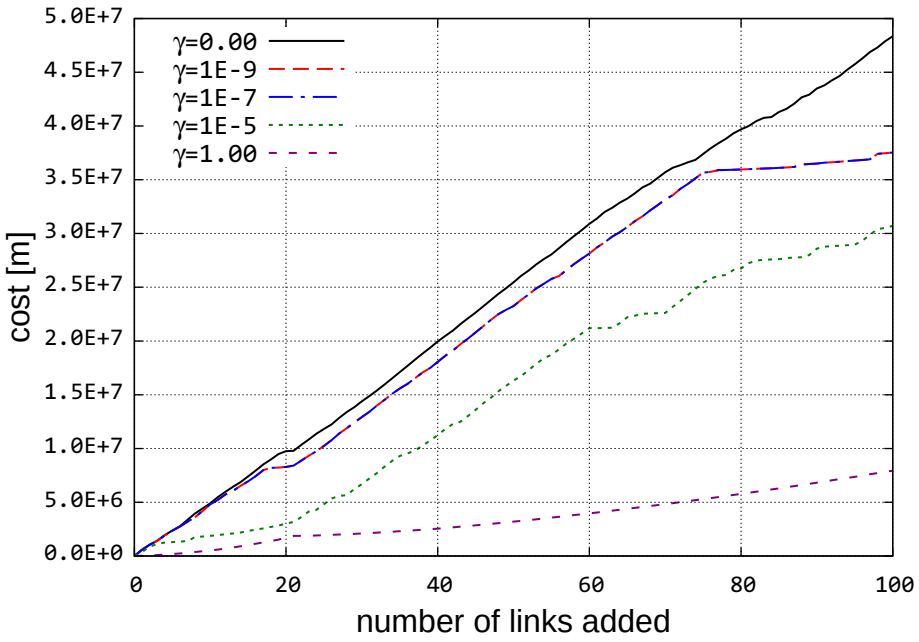

Figure 5.5: Cost incurred with adding links for Sprint physical topology

adding 100 links. On the other hand, if cost is not considered (i.e. $\gamma = 0$) the algebraic connectivity of the Sprint topology can be improved to more than 0.035.

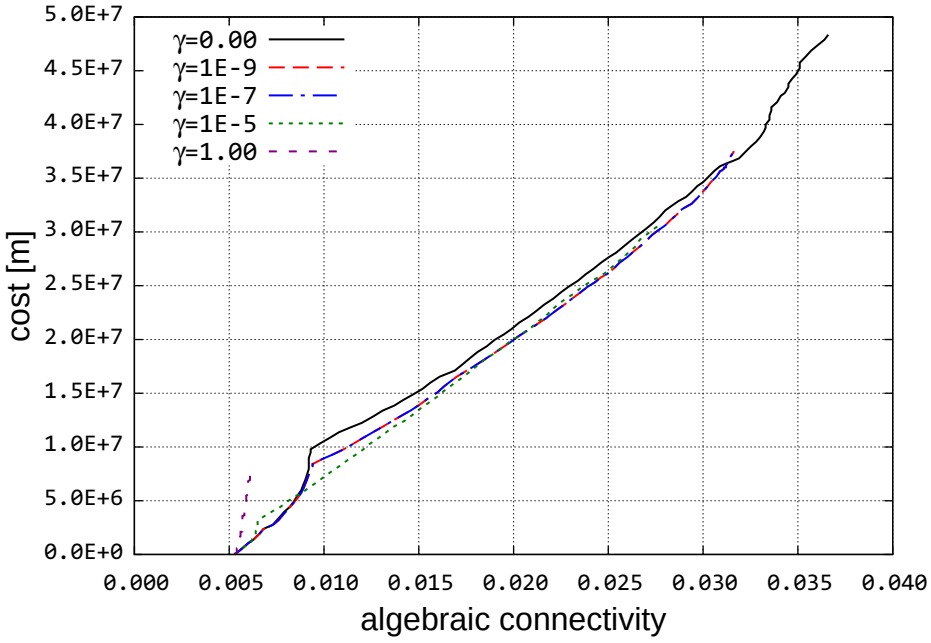

Figure 5.6: Connectivity and cost trade-offs for Sprint physical topology

Here, we have presented and analyzed Sprint physical topology in terms of algebraic connectivity improvement and incurred cost. In addition to the Sprint physical topology improvement plots, both AT&T and Level 3 physical topologies are presented in Appendix B.1.1.

5.2.2 Robustness Evaluation

In this section, we present the results of applying the graph centrality attacks to non- and improved graphs by removing 50 nodes from each graph. From the improved graphs, we select the graphs generated using $\gamma = \{0, 10^{-7}, 1\}$ since they represent the lowest, middle, and highest γ values. The improved graphs when $\gamma = 0$ are expected to be the most resilient since new links are selected purely to improve algebraic connectivity with no cost consideration. The improved graphs when $\gamma = 10^{-7}$ are expected to be the second most resilient graphs since new links are selected to improve algebraic connectivity while favoring the least cost links with a threshold related to 10^{-7}. The improved graphs when $\gamma = 1$ are expected to be the least resilient graphs since the new links are selected to purely minimize the total cost.

The node betweenness attack is consistently the most destructive, since flow robustness decreases faster than the other two centrality attacks as shown in Figures 5.7, B.19, B.22, B.25, and B.28. The second most destructive centrality attack among the three is the closeness attack since it shows a higher impact on the flow robustness. The least destructive attack is the degree attack as it yields the lowest impact on flow robustness. Table 5.2 shows the sum of flow robustness values, which represent the area under the flow robustness curve after 50 nodes are removed. The $a(G)$-improved graphs are more resilient than non-improved graphs because they have 100 additional links.

For AT&T non- and improved graphs, the results of applying three centrality attacks

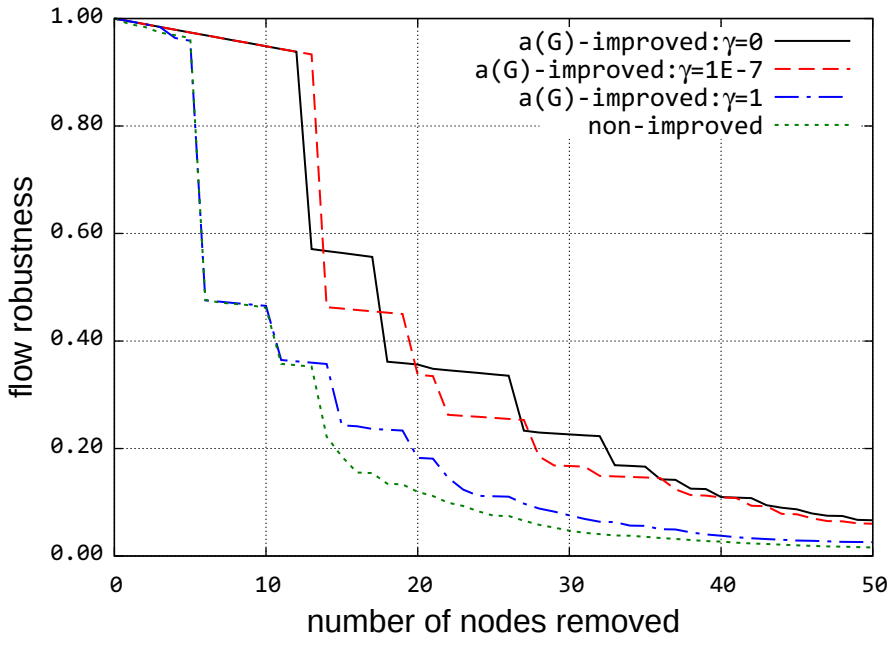

Figure 5.7: AT&T betweenness-based attack

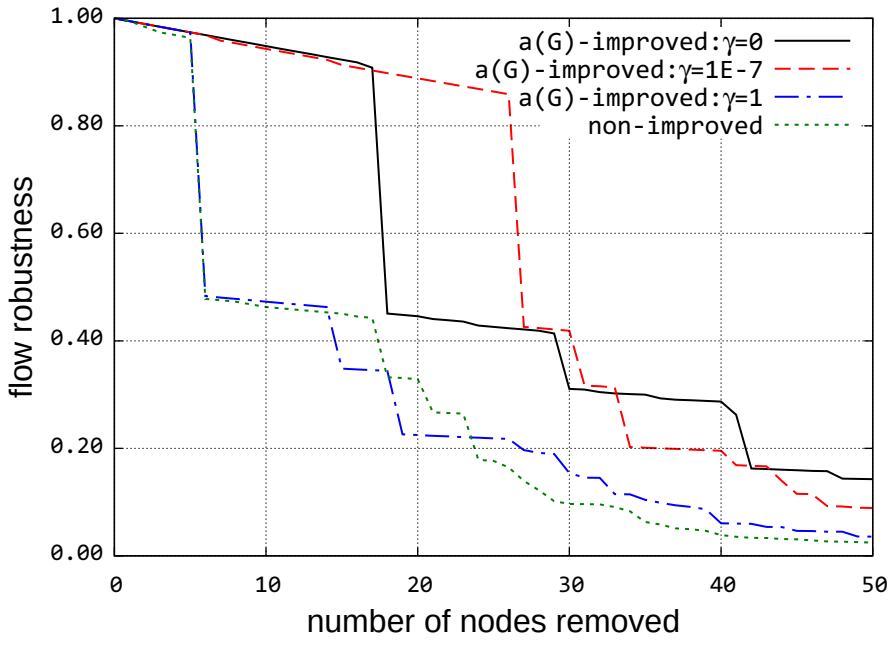

Figure 5.8: AT&T closeness-based attack

are shown in Figures 5.7, 5.8, and 5.9. For the betweenness attack on AT&T non- and improved graphs, we observe that the $a(G)$-improved graph when $\gamma = 0$ has the highest

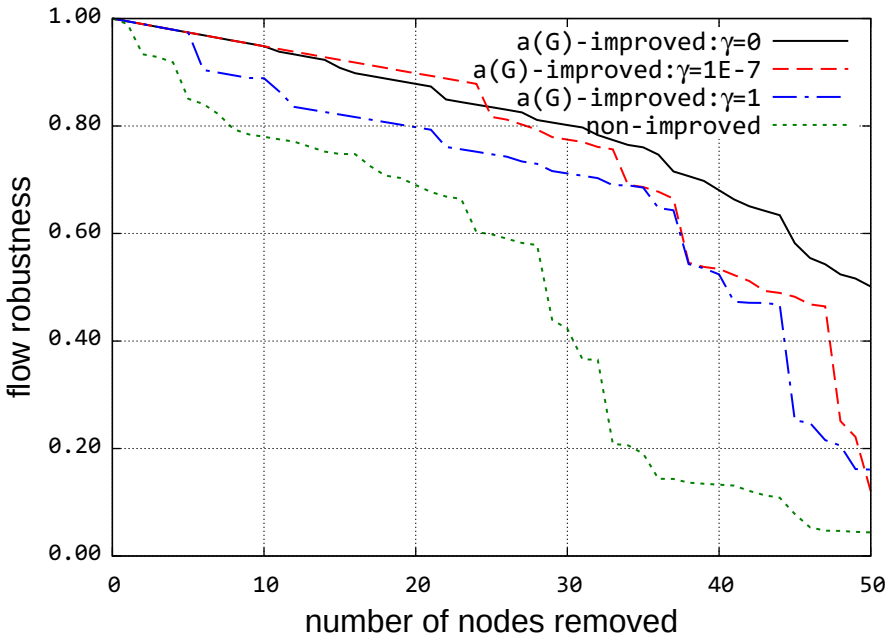

Figure 5.9: AT&T degree-based attack

sum of flow robustness of 21.90 as shown in Table 5.2. Next, the $a(G)$-improved graph when $\gamma = 10^{-7}$ comes second in terms of flow robustness with an insignificant difference of 21.14. Improved graphs when $\gamma = 1$ and non-improved graphs have the lowest flow robustness values of 13.01 and 11.70 respectively. For the closeness attack, the $a(G)$-improved graph when $\gamma = 10^{-7}$ has the highest flow robustness value of 30.31 and the second highest is when $\gamma = 0$ with a flow robustness sum of 27.32. Similar to betweenness attack results, improved graphs when $\gamma = 1$ and non-improved graphs have the lowest flow robustness values of 15.60 and 15.15 respectively. For the degree attack, the $a(G)$-improved graph when $\gamma = 0$ has the highest flow robustness value of 41.34 and the second highest is when $\gamma = 10^{-7}$ with a flow robustness sum of 38.89. Similar to betweenness attack results, improved graphs when $\gamma = 1$ and non-improved graphs have the lowest flow robustness values of 15.60 and 15.15. Given the previous flow robustness for all the attacks on AT&T non- and improved graphs, we see that $a(G)$-improved graph when $\gamma = 0$ is more resilient than the other γ values. However, in some cases, the $\gamma = 10^{-7}$

yields very similar flow robustness results to the improved graphs using $\gamma = 0$. In these cases, feasible graphs can be selected based on the available budget. For example, for the betweenness attack on AT&T, the flow robustness sum difference between $\gamma = 0$ and $\gamma = 10^{-7}$ is $21.90 - 21.14 = 0.76$, which is insignificant. On the other hand, the cost difference between the two is about $5.3 \times 10^7 - 4.0 \times 10^7 = 1.3 \times 10^7$ m, which is a significantly high cost. At this point, it depends on the user to decide if this additional flow robustness is worth 1.3×10^7 m or link cost.

Using the same method, we study the flow robustness values for the other providers' non- and improved graphs for each centrality attack presented in Table 5.2. From these results, we can see very clearly the same pattern in AT&T non- and improved graphs. The graphs improved using $\gamma = 0$ are the most resilient to any centrality attacks for the examined physical graphs. However, this is not always the case; we have four cases where $\gamma = 10^{-7}$ yields more flow robustness sums than $\gamma = 0$. The first case happens in the AT&T graphs with closeness attack and the other three cases happen due to the degree attack on the three providers: Sprint, Internet2, and CORONET. By looking at the corresponding algebraic connectivity values for each case, we see that the algebraic connectivity values are higher for $\gamma = 0$ even though $\gamma = 10^{-7}$ have higher flow robustness sums for these cases. However, for the other 11 cases the flow robustness sums are higher for graphs with higher algebraic connectivity values.

We apply our algorithm, described in Section 4.1.1, to physical-level topologies of the several backbone providers. The results showed trade-offs between improving algebraic connectivity and minimizing cost, from which a cost-efficient set of link addition can be chosen based on the value of γ. Moreover, we applied centrality-based attacks on the non- and improved graphs and study their resilience in terms of flow robustness. We showed that graphs with higher algebraic connectivities have mostly higher flow robustness values, which means that they are more resilient. The complete set of plots

Table 5.2: Algebraic connectivity improvement evaluation via flow robustness

Provider	**Improvement Method**	**Betweenness attack**	**Closeness attack**	**Degree attack**
AT&T	non-improved	11.70	15.15	25.66
	$a(G)$-improved: $\gamma = 0$	21.90	27.32	41.34
	$a(G)$-improved: $\gamma = 10^{-7}$	21.14	30.31	38.89
	$a(G)$-improved: $\gamma = 1$	13.01	15.60	35.55
Level 3	non-improved	5.68	7.15	7.30
	$a(G)$-improved: $\gamma = 0$	13.81	16.58	23.32
	$a(G)$-improved: $\gamma = 10^{-7}$	10.96	12.22	21.33
	$a(G)$-improved: $\gamma = 1$	7.96	10.61	21.05
Sprint	non-improved	8.46	11.10	14.31
	$a(G)$-improved: $\gamma = 0$	14.50	20.64	29.39
	$a(G)$-improved: $\gamma = 10^{-7}$	13.87	16.38	31.68
	$a(G)$-improved: $\gamma = 1$	10.41	15.10	26.75
Internet2	non-improved	4.09	5.00	4.71
	$a(G)$-improved: $\gamma = 0$	8.98	10.20	15.47
	$a(G)$-improved: $\gamma = 10^{-7}$	8.74	9.23	15.94
	$a(G)$-improved: $\gamma = 1$	8.12	8.37	13.84
CORONET	non-improved	7.43	7.84	9.87
	$a(G)$-improved: $\gamma = 0$	10.82	17.23	18.60
	$a(G)$-improved: $\gamma = 10^{-7}$	10.39	14.38	18.61
	$a(G)$-improved: $\gamma = 1$	8.70	10.62	19.60

for evaluating algebraic connectivity improvement for the five physical-level topologies is presented in Appendix B.1.2.

5.3 Improvement of Path Diversity

In this section, we use CORONET [97, 98], Internet2 [95], and Level 3 [110] fiber-level topologies presented in Section 3.1.3 for evaluating our path diversity improvement al-

gorithm. We apply the improvement algorithm on three realistic backbone networks and study the results. Then, we apply three centrality-based attacks to the resulting improved and non-improved graphs and show how the robustness changes for each graph.

5.3.1 Improvement Analysis

We apply the improvement algorithm on three realistic backbone service provider graphs and study the TGD improvement and the cost incurred for each graph as we add 20 links. We vary the value of path diversity k and hop-count h while λ is set to 0.5.

Varying the hop-count threshold h

The hop-count threshold h is a parameter that controls the length of the shortest path returned by the k diverse algorithm introduced in Section 4.2.1. Therefore, to obtain the optimal diverse paths, the value of h should be larger or equal to the diameter of the graph in order to examine all possible paths in the graph. However, for large graphs, large values of h may take an impractical time to calculate. Here, we apply the algorithm with several values of hop count thresholds $h = \{5, 10, 15\}$ while the value of k is set to 12. These values show how varying the parameter h affects the value of TGD.

Figure 5.10 depicts the results of each hop count threshold, which shows the TGD improvement as 20 links are added to the Internet2 topology. As the hop count threshold increases, the size of the candidate set also increases, which in turn increases the probability of having a higher EPD value. As a result, a 5 hop-count threshold has the lowest TGD while 10 and 15 have the median and highest TGD as shown in Figure 5.10.

As the hop-count threshold increases, the cost of adding links to Internet2 does not follow the same pattern of TGD improvement. We observe that the 10 hop-count starts as the lowest costly approach for the first 7 links while toward the end, the 15 hop-count becomes

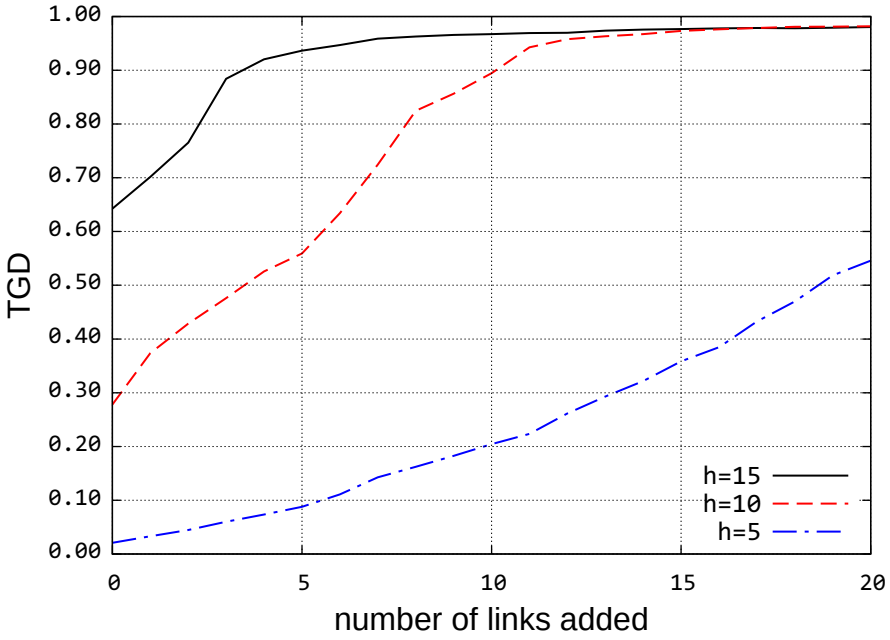

Figure 5.10: Internet2 TGD improvement

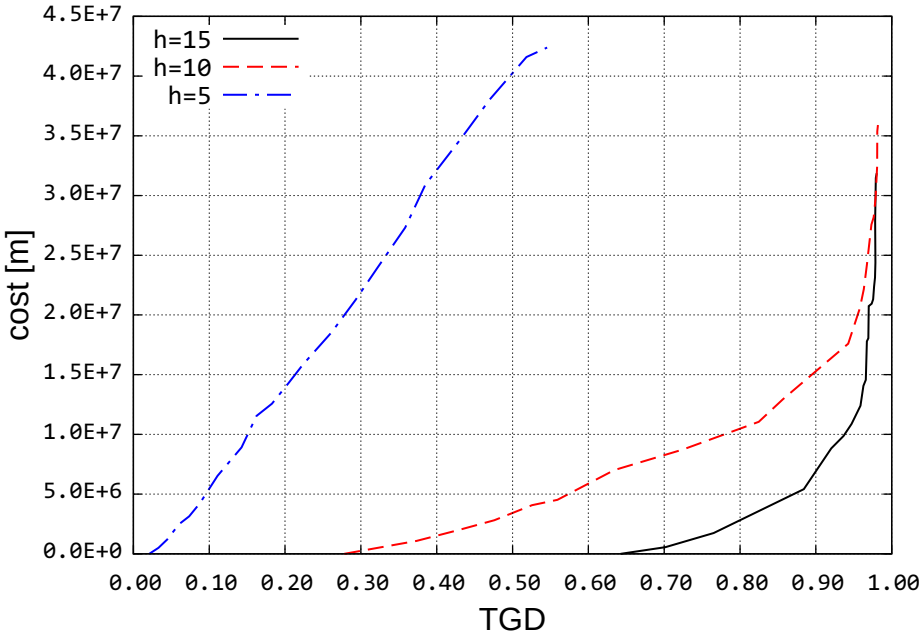

Figure 5.11: Internet2 cost and TGD

as the lowest costly approach as shown in Figure 5.12. This indicates that changing the hop-count parameter does not have direct affect on the incurred cost. Thus, the cost

incurred depends on the initial topological properties such as the number of nodes and links, average degree, and node locations. However, the cost needed to achieve a certain TGD for Internet2 topology is shown in Figure 5.11, which shows that as the hop count threshold increases, the cost to achieve a certain TGD in general decreases. We expect this outcome to occur because a higher hop count threshold starts from a higher TGD for the same reason mentioned earlier.

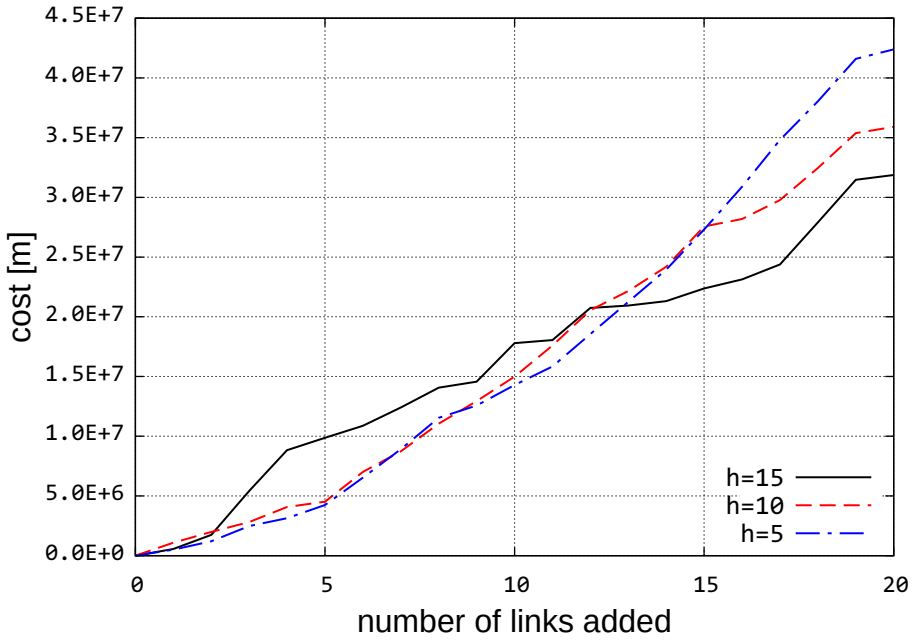

Figure 5.12: Internet2 cost incurred

We have presented relevant plots to study the hop count impact for Internet2 physical-level topology. The complete set of plots for evaluating algebraic connectivity improvement for the other physical-level topologies is presented in Appendix B.2.1, which show similar results to Internet2. Among the three networks, we observe that the cost needed to achieve a certain TGD decreases as the hop count threshold increases.

Varying the number of diverse paths k

The number of diverse paths k is a parameter that controls the number of the most diverse paths returned by the k diverse algorithm introduced in Section 4.2.1. The value of k depends on the application of the graph. For example, if the provider uses a multipath routing protocol with a threshold for the number of multipaths used, k can be chosen to match that parameter for accurate path diversity. Choosing a high value of k does not have a processing complexity penalty similar to choosing a higher value of h.

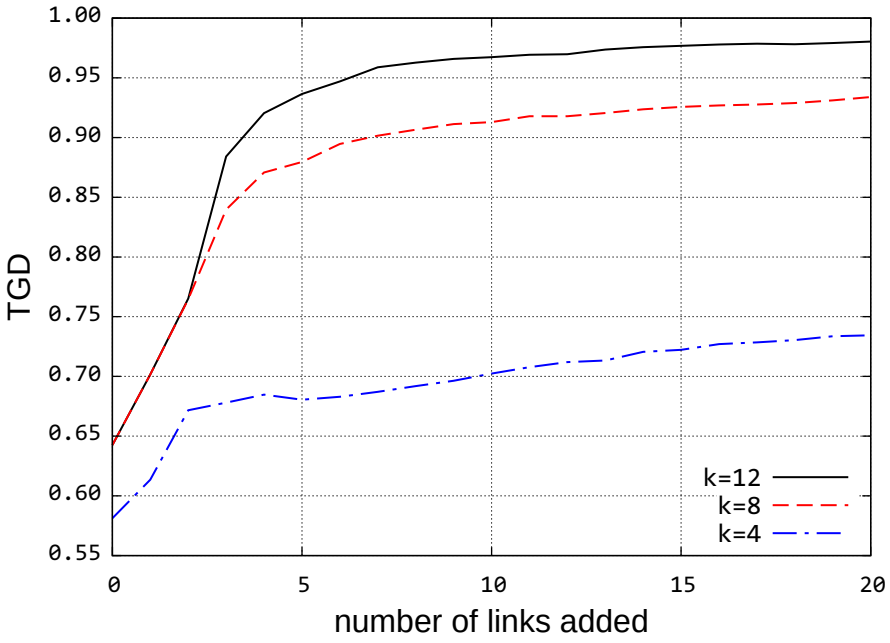

Figure 5.13: Internet2 TGD improvement

We apply the algorithm with several values of the number of diverse path threshold $k = \{4, 8, 12\}$ while the value of h is set to 15. As the value of k increases, the length of diverse path set also increases, which in turn increases effective path diversity for a given pair of nodes. Consequently, as the value of k increases, the corresponding TGD increases as shown in Figure 5.13 for Internet2 topology. However, the length of the diverse paths set is actually m, which does not increase as the maximum diversity of the

remaining paths is zero as mentioned in Section 2.2.4. For this reason, the two k values 8 and 12 have similar outcomes as depicted in Figure 5.13. The cost incurred depends on the initial topological properties as shown in Figure 5.14. The cost needed to achieve a certain TGD for the Internet2 graph is shown in Figure 5.15, which shows that as the value of k increases, the cost to achieve a certain TGD decreases as long as the increasing the value of k actually increases the value of m.

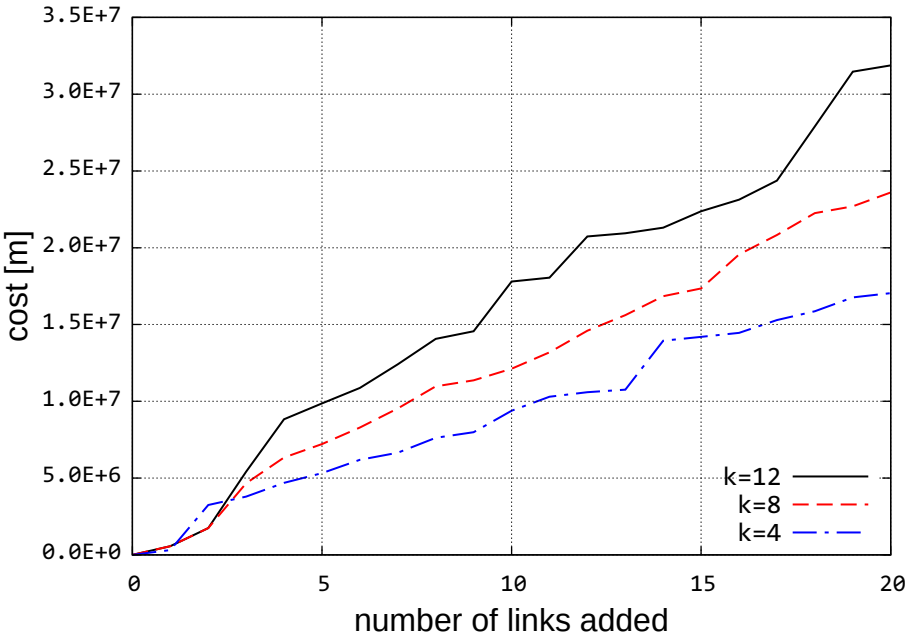

Figure 5.14: Internet2 cost incurred

We have presented relevant plots to study the effect of a k diverse path parameter for Internet2 physical-level topology. The complete set of plots for evaluating algebraic connectivity improvement for the other physical-level topologies is presented in Appendix B.2.2. The results of all three networks showed that the cost to achieve a certain TGD decreases the value of k increases.

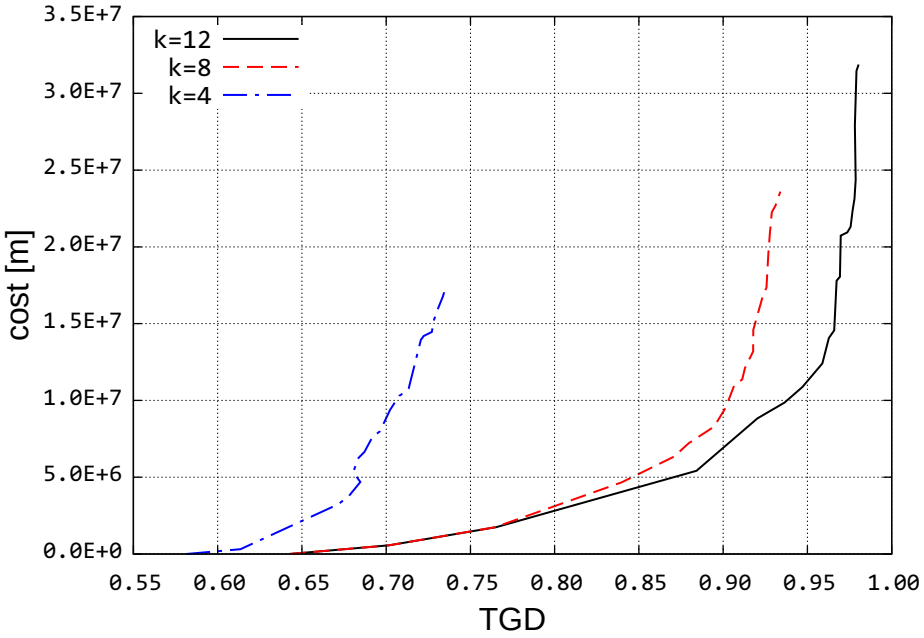

Figure 5.15: Internet2 cost and TGD

5.3.2 Robustness Evaluation

In this section, we present the set of attacks used to evaluate the flow robustness (cf. Section 2.2.1) of the resulting improved and non-improved topologies. Then, we apply these attacks and display the results.

Graph centrality attacks

We use a graph-theoretic model to attack a given graph and show how its flow robustness changes after each node removal. In this research, we use three centrality metrics: node betweenness, closeness, and node degree. Thus, we have three attack models, in which the node with the highest centrality is removed. The node-betweenness attack targets the node through which the highest number of shortest paths pass. The node-closeness attack targets the closest node to all the other nodes in terms of hop count. The highest node degree attack targets the node with the highest number of neighbors. The list of

removed nodes is determined adaptively for each attack model. This indicates the node centrality values are calculated after each node is removed and the highest is selected to be the next node for removal. This is done repeatedly until all nodes are selected. The adaptive removal of nodes gives a more effective selection for the highest centrality than the non-adaptive removal, in which the highest targeted number of nodes are selected based on a single evaluation.

Lowest degree improvement

For comparison purposes, we introduce an intuitive improvement algorithm to improve the connectivity of a given graph via adding links to the smallest degree nodes. This algorithm adds one link repeatedly until a number of links requested by the user is added. On each iteration, one end of the link is connected to the least degree node and the other end is connected to the next least degree node. If there are multiple least-degree candidate links, the least-cost link is selected to be added.

Robustness evaluation results

In this section, we show the results of applying graph centrality-based attacks to path diversity improved (PD-improved), lowest degree improved (LD-improved), and non-improved topologies. For the set of PD-improved graphs, we choose the set generated using the hop count threshold $h = 15$ and the number of diverse path threshold $k = 12$ because both have yielded the diverse results as shown in Sections 5.3.1 and 5.3.1. For each graph, we apply the attack by removing half of its original number of nodes. The flow robustness is calculated after each node removal. The node-betweenness attack has the highest negative impact on flow robustness because it targets the most vital nodes in the Internet2 graph as shown in Figures 5.16 through 5.18. The second highest negative impact on flow robustness is done by the highest-closeness node attack since the target

node has the highest-closeness to all the other nodes in terms of hop count. The least negative impact on flow robustness comes from the highest-degree node since it has a higher number of neighbors but is not necessarily used by the highest number of shortest paths.

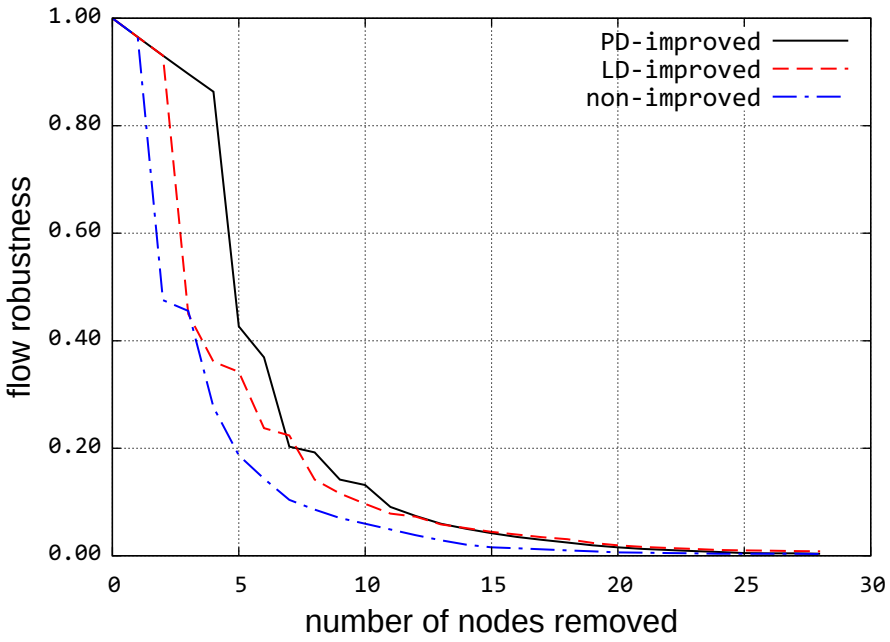

Figure 5.16: Robustness of Internet2 against betweenness-based attack

Both PD- and LD-improved graphs are more resilient than non-improved graphs because they have 20 additional links. For example, the total flow robustness of the PD-improved Level 3 graph under the betweenness attack is 10.1 while it is 6.5 for the LD-improved and 5.7 for the non-improved graphs. Among the three provider graph analyses, the PD-improved graphs are more resilient than the LD- and non-improved graphs for betweenness and closeness attacks. For degree-based centrality attacks, LD-improved graphs have higher flow robustness since links are added to the lowest degree nodes, which are targeted *the least* as shown in Figure 5.18. Therefore, the links connected to the lowest degree nodes using LD-improvement contribute more to flow robustness than links added using PD-improvement during the degree-based attack.

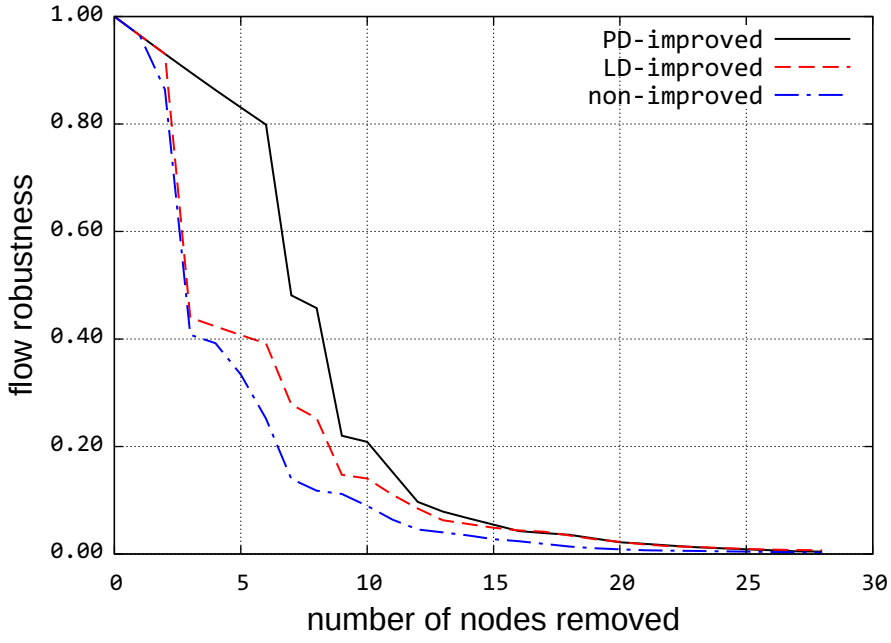

Figure 5.17: Robustness of Internet2 against closeness-based attack

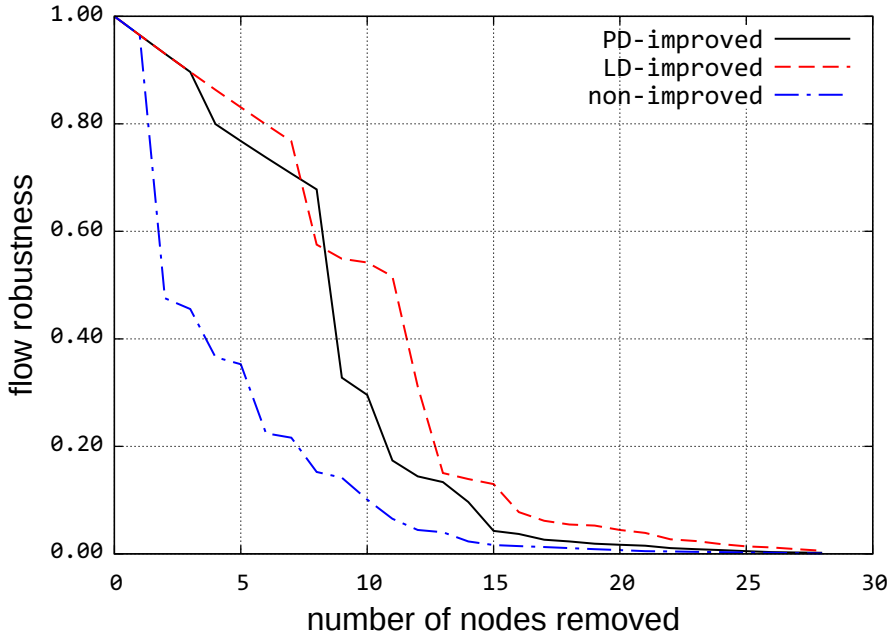

Figure 5.18: Robustness of Internet2 against degree-based attack

PD-improved graphs have higher flow robustness in most physical-level graphs because in PD-improved graphs links are added to increase the number of diverse paths, the most

among all communicating nodes in the graph. Thus, when a node is removed from a PD-improved graph, it slightly affects the other communicating nodes since they have more alternative paths to reroute their communication. In contrast, when a node is removed from an LD-improved graph, the other communicating nodes are more affected because they have fewer alternate paths among the communicating nodes. This is because LD-improvement adds links based on the objective of increasing the connectivity of the lowest degree node rather than increasing the number of diverse paths.

In this section, we have presented relevant plots to evaluate non- and improved Internet2 physical-level topology. The complete set of plots for evaluating PD-improvement for the other physical-level topologies is presented in Appendix B.2.3. The other physical-level topologies showed similar results. For all networks, using the flow robustness graph metric, the path diversity improved graphs were compared to both lowest degree non- and improved graphs as they were attacked using node removal based on highest node centrality graph metrics. The path diversity improved graphs showed better resilience to these attacks compared to the lowest degree non- and improved graphs. PD-improved graphs have higher flow robustness in most physical-level graphs because in PD-improved graphs, links are added to increase the number of diverse paths the most among all communicating nodes in the graph. Thus, when a node is removed from a PD-improved graph, it slightly affects the other communicating nodes since they have more alternative paths to reroute their traffic.

5.4 Improvement via Balancing Centrality

In this section, we apply our improvement algorithm on the Internet2, CORONET, and Level 3 physical-level topologies presented in Section 3.1.3. Then, we apply three

centrality-based attacks to the resulting improved and non-improved graphs and show how the robustness changes during each attack.

5.4.1 Improvement Analysis

In this section, we apply our improvement algorithm on three backbone service-provider graphs and study the improvement and the cost incurred for each graph as links are added. The budget constraint for all graphs is 5×10^7 meters. The budget is measured as the sum of the length of the added links.

Betweenness improvement analysis

We apply our algorithm on the three physical graphs while the objective function is set to minimize betweenness variance. In other words, links are added to uniformly utilize all nodes in terms of forwarding traffic. The betweenness variance changes while links are added is shown in Figure 5.19. For all providers, the betweenness variance starts around 7×10^{-3}, which is the initial variance. While adding the first 20 links the variance decreases dramatically to 1×10^{-3}. The first 20 added links contribute greatly to minimizing the variance, and the remaining links do not have any significant impact on minimizing the betweenness variance. Therefore, the variance slowly decreases as the rest of the links are added. The costs incurred as links are added for all providers are shown in Figure 5.20. The cost is increasing at a similar pace for all providers but their slopes are different. The cost of adding the first 20 links increases more than after the 20th link because these links are selected regardless of their high cost since they contribute greatly to minimizing the betweenness.

Level 3 has the lowest slope, which provides the highest number of added links. This is because Level 3 has the largest number of nodes, which gives more affordable candidate

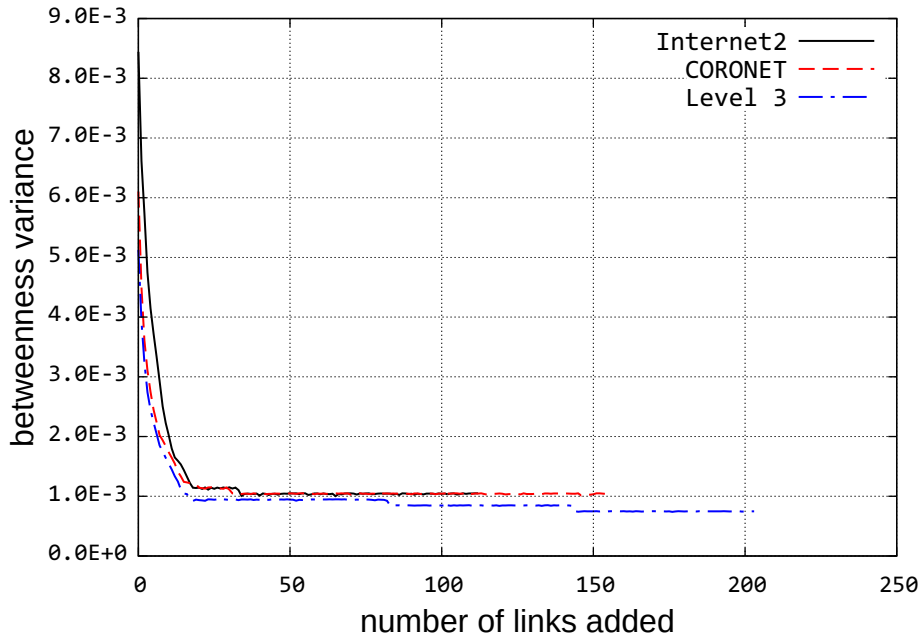

Figure 5.19: Minimizing variance of node betweenness

links to select from. On the other hand, the incurred cost while adding links to Internet2 increases faster than the other two because it has a lower number of links, which in turn yields a lower number of candidate links. As a result, expensive links are selected, which consumes the budget more quickly.

Closeness improvement analysis

While selecting the objective function to minimize the variance of the node closeness of the graph, we apply our algorithm to the three graphs. In other words, links are added to make the shortest-path distance between all the nodes more uniform. The variance changes while links are added is shown in Figure 5.21. The variance of closeness values are different for the three providers. For Level 3 the closeness variance decreases overall. On the other hand, for Internet2 and CORONET, the variance decreases while adding the first several links, and then it fluctuates around 5.5×10^{-4}. However, why the fluctuations happen *only* for closeness-based improvement is not known and the reasons

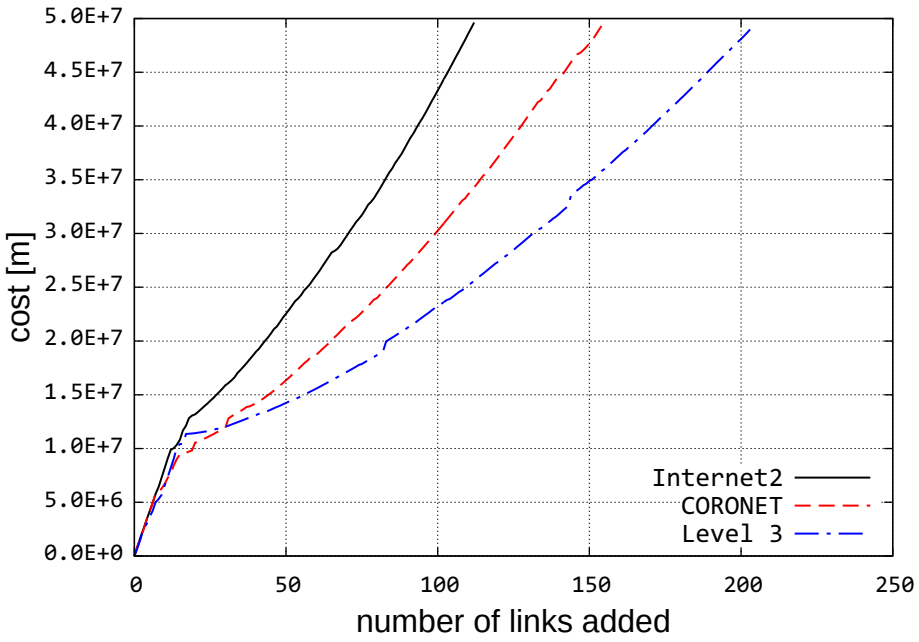

Figure 5.20: Cost of improving node betweenness

for the occurrence of this phenomenon will be the subject of future work. The costs incurred as links are added for all providers are shown in Figure 5.22. Here, the costs are similar with no phase changes since the added links do not have a significant contribution to minimizing closeness variance for Internet2 and CORONET. For Level 3, the cost exhibits the same increase of the other two providers overall. However, while adding the 60th link, we observe a small jump in the cost, which corresponds to a significant decrease in the variance for the same link. This is because the algorithm selects links that minimize the variance regardless of their cost, which in this case is higher than other selected links.

Degree improvement analysis

Here, we apply our algorithm to the three physical graphs while the objective function is set to minimize degree variance in order to have uniform node degree graph. The variance changes while links are added is shown in Figure 5.23. For all the providers, the degree

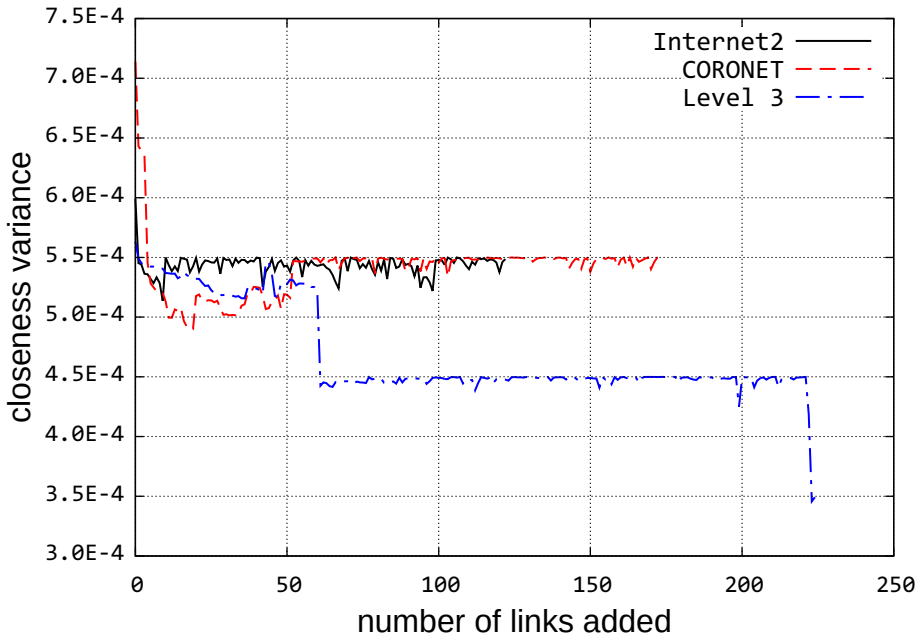

Figure 5.21: Minimizing variance of node closeness

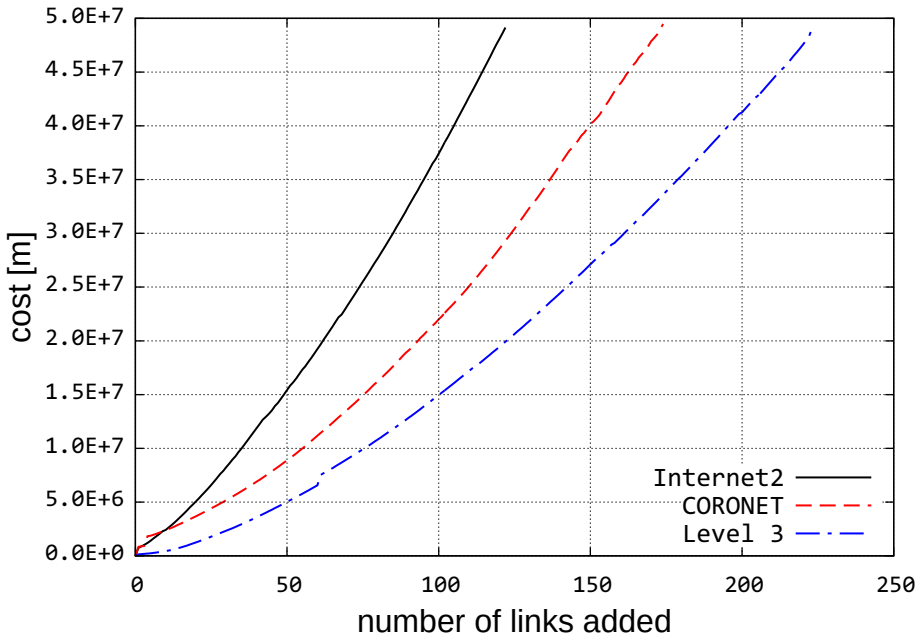

Figure 5.22: Cost of improving node closeness

variance starts from different initial values but they are *not monotonically decreasing, but rather oscillating.* To explain this phenomenon, let us start with uniform node degree

graphs, where each node has a degree of k, which gives a graph with zero degree variance. To add links to the graph, the variance has to increase no matter where the links are placed. The variance increases until it reaches a top point and then it decreases to reach the zero where the graph has a $k+1$ node degree. If the number of links needed to increase k to $k+1$ is x, then the top points must be around adding the next $x/2$ links. Now, we can observe that the degree variance in Figure 5.23 does not reach zero for any providers. This is because long links cannot be selected. The costs incurred as links are added for all providers are shown in Figure 5.24. For all providers, the cost increases overall with similar pattern of phase changes. For example, Level 3 link addition cost increases at the same rate. Then, the rate increases until the 27th link is added, which slows the rate of cost increase. The point where the cost slows down corresponds to the point where the degree variance is at a minimum. This happens because at this point, the graph has the maximum number of candidate links to get the next minimum degree variance of the graph. Hence, the lowest cost link is selected. However, after adding a few links with the low cost, the remaining links in the candidate set are all expensive. Therefore, the algorithm has to select one of these links, and the links added before reaching the lowest variance are more expensive than links selected after passing this point.

Improvement method vs. number of added links

Using the improvement results shown in Figures 5.19, 5.21, and 5.23, we observe that while limiting the budget to a constant value for all the graphs, the actual number of added links for a given graph differ based on the used improvement method. For example, for the Internet2 graph, the numbers of added links using betweenness, closeness, and degree are: 112, 122, and 112 respectively. For the CORONET graph, the numbers of added links using betweenness, closeness, and degree are: 154, 174, and 141. For the Level 3 graph, the numbers of added links using betweenness, closeness, and degree are:

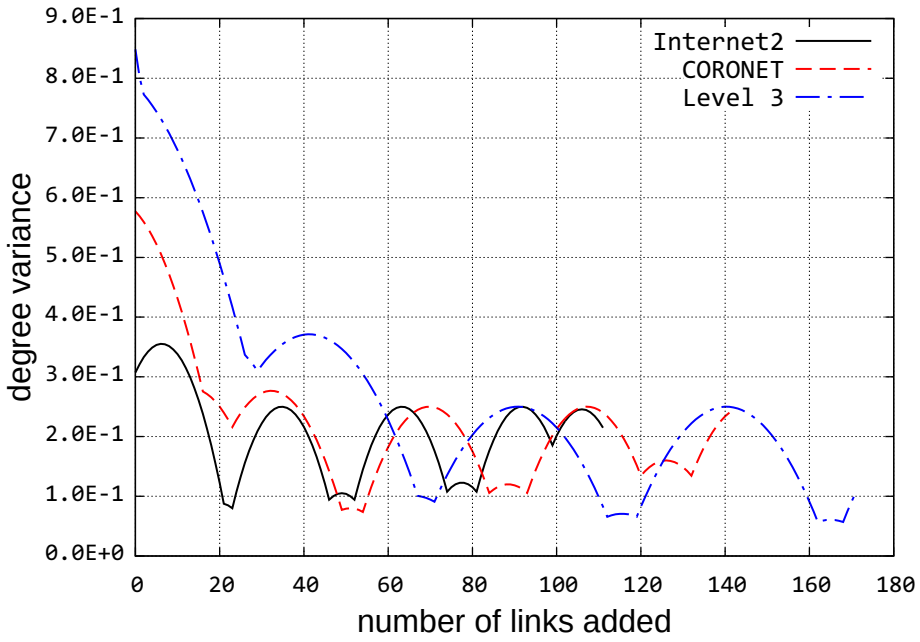

Figure 5.23: Minimizing variance of node degree

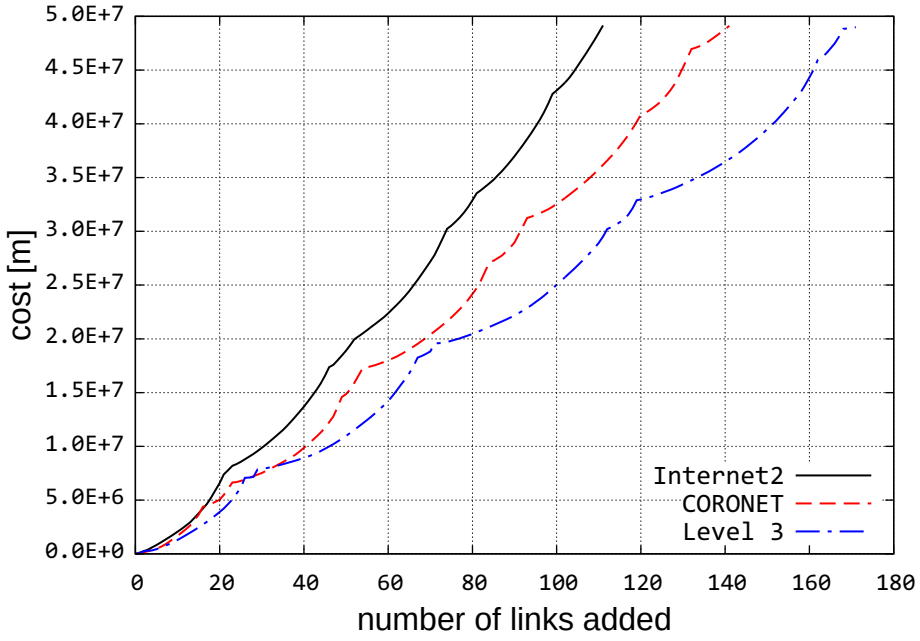

Figure 5.24: Cost of improving node degree

203, 224, and 171. From these numbers, we observe that closeness improvement methods always yield the highest number of added links, which implies that they tend to select

shorter links. On the other hand, both betweenness and degree improvement yield a fewer number of links. The degree-based improvement method yields the lowest number of links added with small improvement differences with respect to betweenness-based method.

5.4.2 Robustness Evaluation

Here, we show the results of applying the graph centrality attacks to non- and improved graphs while computing the flow robustness of the graph as nodes are removed during the attack, which causes the removal of 50 nodes from each graph.

The sum of flow robustness values are shown in Table 5.3. By comparing the sum of flow robustness values of the non- and improved graphs, we can see that the betweenness attack overall yields lower flow robustness values, which implies that the betweenness attack is the most destructive attack on these physical graphs. Next, using the same approach, the closeness attack is second, and the degree attack is the least destructive attack.

The results of applying three centrality attacks on Internet2 non- and improved graphs are shown in Figures 5.25, 5.26, and 5.27. For the betweenness attack on Internet2 non- and improved graphs, we observe that the degree-improved graph has the highest value of flow robustness of 8.68. Furthermore, the betweenness improved graph comes second in terms of flow robustness with a very small difference of 8.65. Even though the closeness-improved graph has the highest number of added links, it yields the lowest flow robustness among the improvement methods for the Internet2 graph as shown in Table 5.3. For the closeness attack, the betweenness-improved graph outperforms the other methods with a flow robustness value of 12.99, while closeness and degree flow robustness values are 10.28 and 8.86. For the degree attack, the degree-improved graph again has the highest

Table 5.3: Sum of flow robustness

Provider	**Improvement method**	**Betweenness attack**	**Closeness attack**	**Degree attack**
Internet2	non-improved	4.09	5.00	4.71
	betweenness	8.65	12.99	15.88
	closeness	6.96	10.28	15.48
	degree	8.68	8.86	16.95
CORONET	non-improved	7.43	7.84	9.87
	betweenness	10.43	12.55	19.72
	closeness	8.76	11.66	20.03
	degree	10.60	11.79	21.28
Level 3	non-improved	5.68	8.86	16.95
	betweenness	11.63	15.36	25.81
	closeness	9.56	18.71	21.54
	degree	11.08	12.07	25.62

flow robustness of 16.95. The betweenness and closeness improvement methods follow with flow robustness values of 15.88 and 15.48. By observing all the flow robustness values for the scenarios we study, the degree improvement is more resilient to attacks for Internet2, and the closeness improvement is the weakest method for the same graph. The betweenness improvement method is second but it is closer to the outcome of degree improvement method.

The results of applying three centrality attacks on CORONET non- and improved graphs are shown in Figures 5.28, 5.29, and 5.30. While applying the betweenness attack on CORONET non- and improved graphs, we see that the degree-improved graph has the highest sum of flow robustness of 10.60. Next, the betweenness-improved graph has a similar value of flow robustness of 10.43. Similar to the outcome of Internet2, the closeness has the lowest flow robustness of 8.68. For the closeness attack, the betweenness-improved graph has the highest flow robustness with a value of 12.55. Next, both degree and closeness have similar values of 11.79 and 11.66 respectively. For the degree attack, the

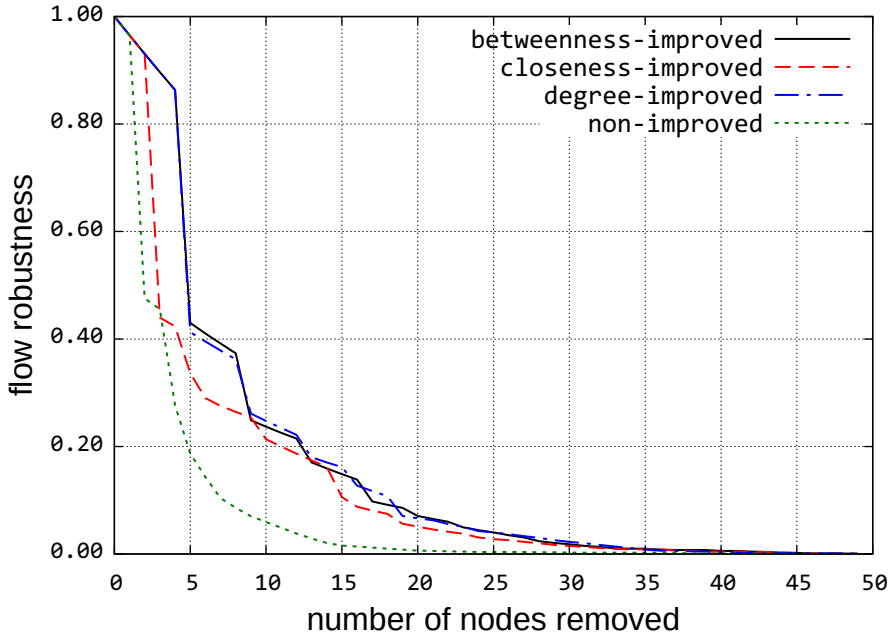

Figure 5.25: Internet2 betweenness-based attack

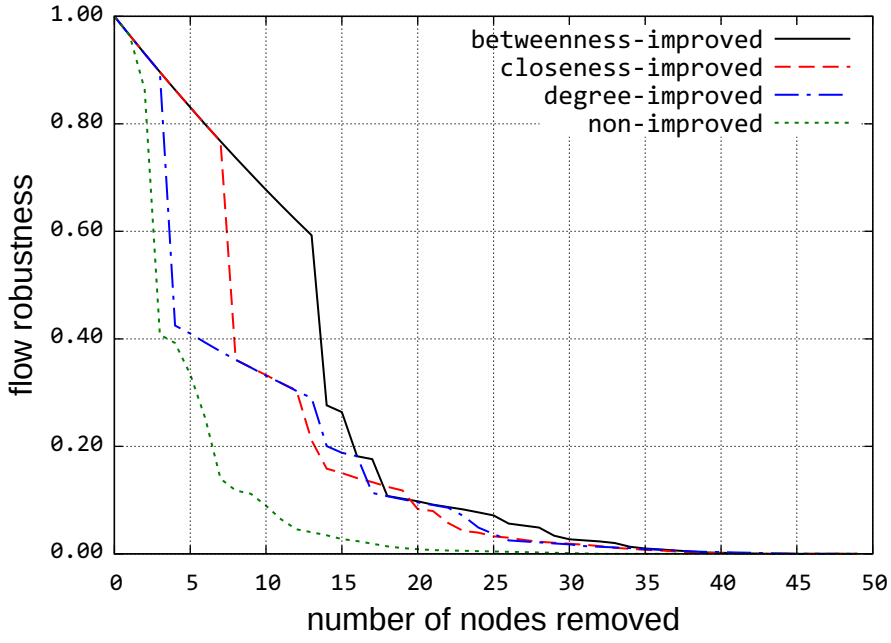

Figure 5.26: Internet2 closeness-based attack

degree-improved graph has the highest flow robustness of 21.28. Closeness comes next with a flow robustness value of 20.03. Finally, the betweenness-improved graph has a flow

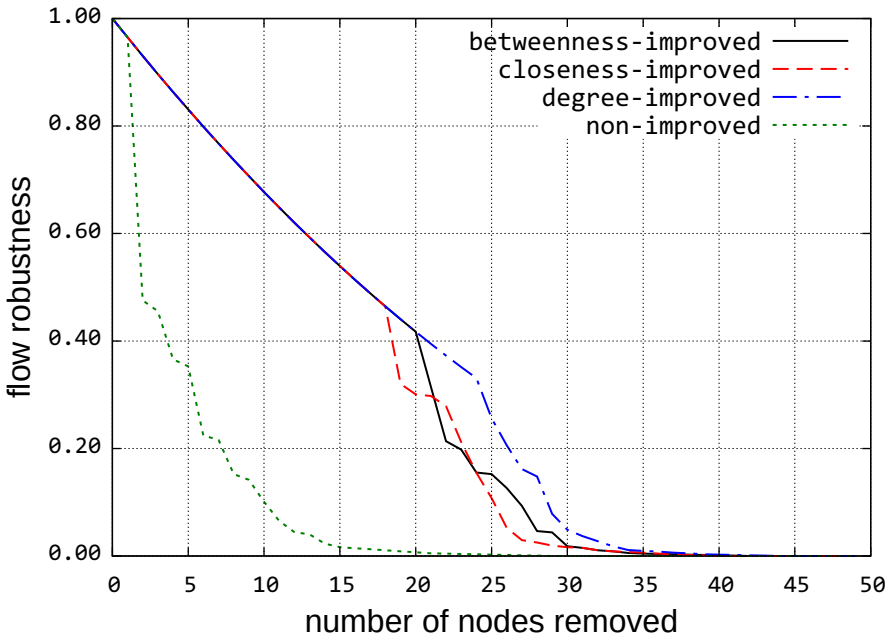

Figure 5.27: Internet2 degree-based attack

robustness value of 19.72. As we can see, similar to Internet2, the degree improvement is the most resilient among the three methods to centrality attacks for the CORONET physical graph.

The results of applying three centrality attacks on Level 3 non- and improved graphs are shown in Figures 5.31, 5.32, 5.33. For the betweenness attack on Level 3 non- and improved graphs, we observe that the betweenness-improved graph has the highest sum of flow robustness of 11.63. The degree-improved graph comes second in terms of flow robustness without much difference of 11.08. Thirdly, the closeness-improved graph has a flow robustness value of 9.56. For the closeness attack, the closeness-improved graph has the highest flow robustness value of 18.71. Next, the betweenness-improved graph yields the second best flow robustness value of 15.36. Last and least is the degree-improved graph with a flow robustness value of 12.07. For the degree attack, the betweenness-improved graph has the best flow robustness value of 25.81. With small variation, the

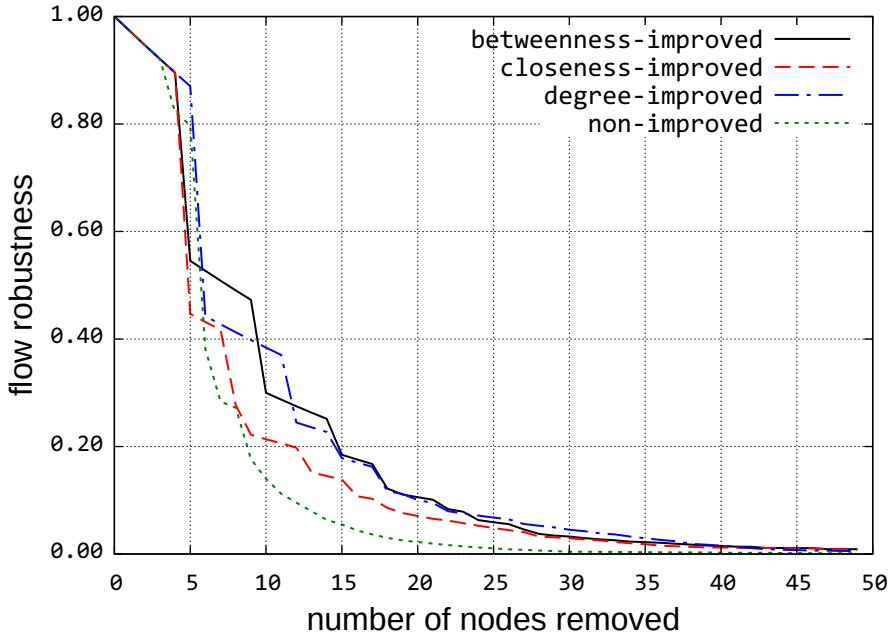

Figure 5.28: CORONET betweenness-based attack

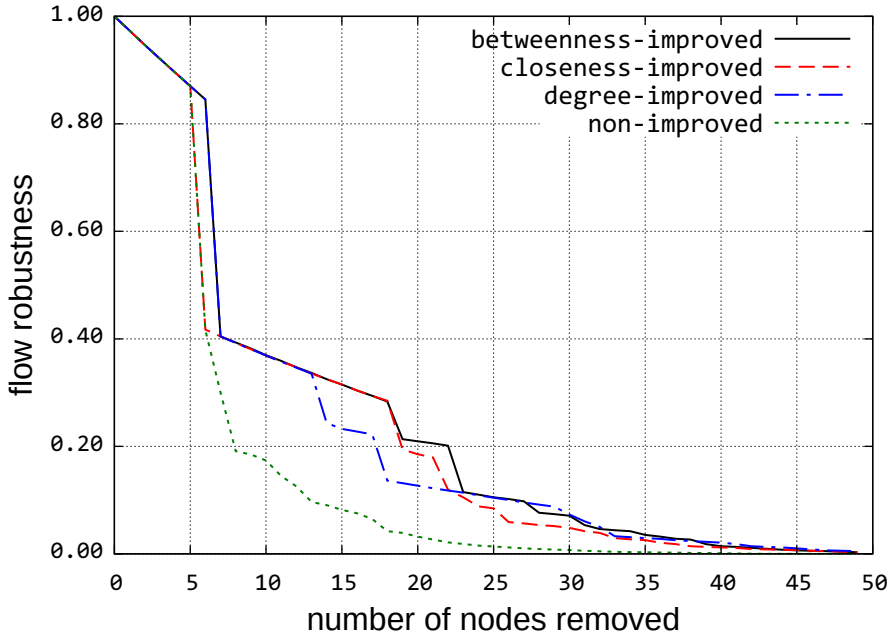

Figure 5.29: CORONET closeness-based attack

degree-improved graph comes in second with a value of 25.62. The closeness-improved graph has the lowest flow robustness value of 21.54. By observing all the flow robustness

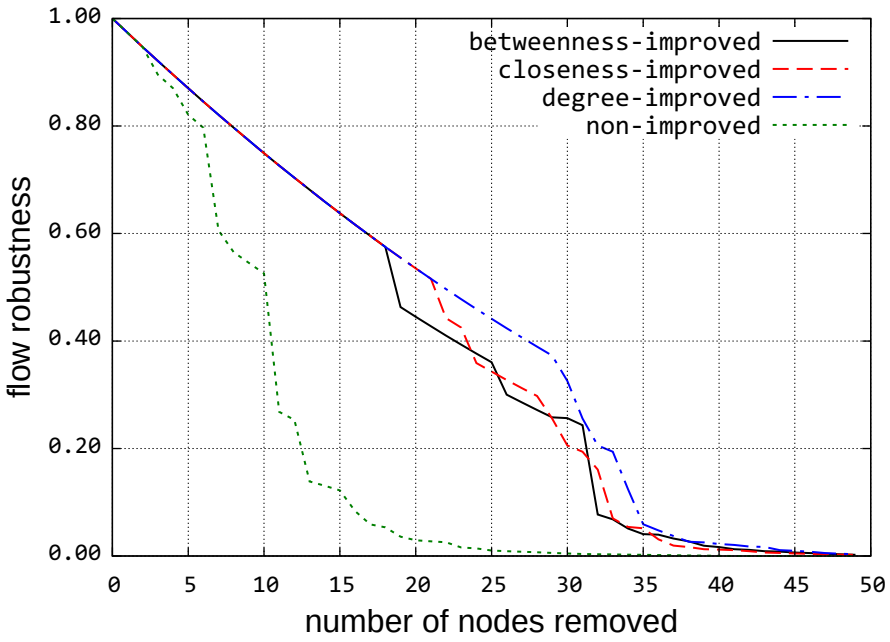

Figure 5.30: CORONET degree-based attack

values for both improvement methods and attacks, we can see that betweenness improvement is more resilient to attacks for Level 3. Then, the degree improvement yields better results than closeness improvement, which is the weakest method for the Level 3 graph.

Finally, we compare the results of the three improvement methods' flow robustness values against the number of added links discussed in Section 5.4.1. We observe that even though the closeness-based improvement consistently yields the highest number of added links, it fails to provide better flow robustness values than both betweenness and degree improvement methods in most attacks. This implies that having a larger number of links in a given graph does not necessarily guarantee a better resilience. Moreover, adding links without a careful improvement of networks may not yield any gain in terms of resilience and performance.

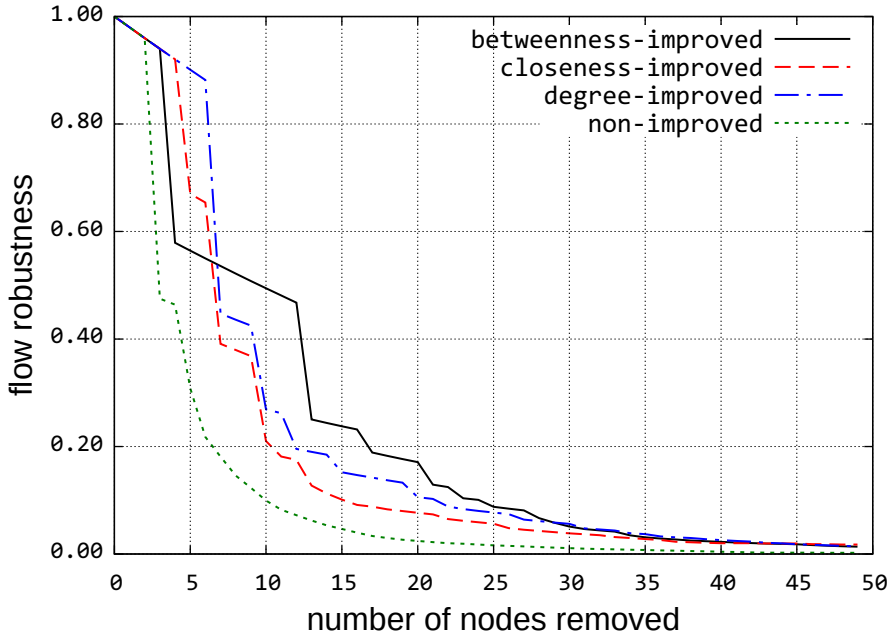

Figure 5.31: Level 3 betweenness-based attack

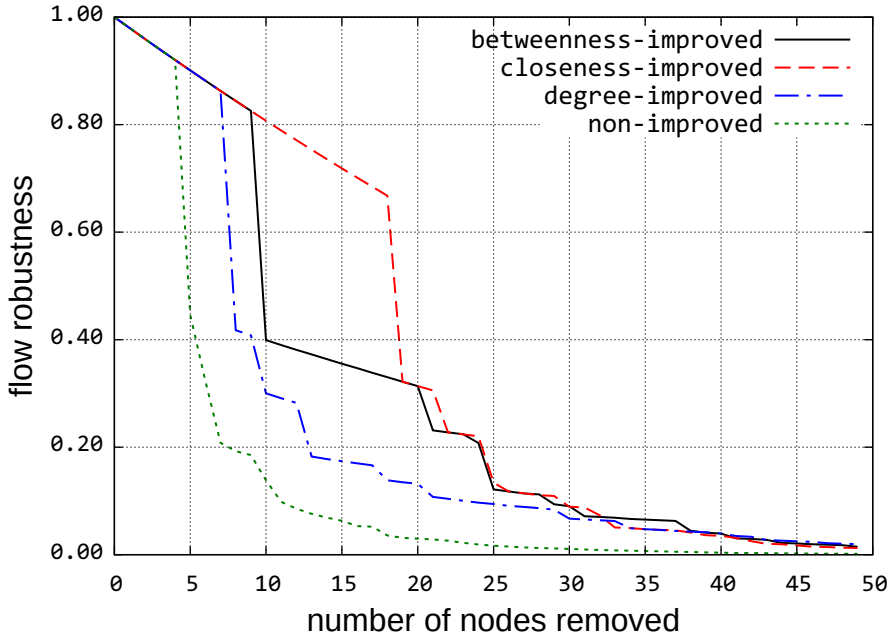

Figure 5.32: Level 3 closeness-based attack

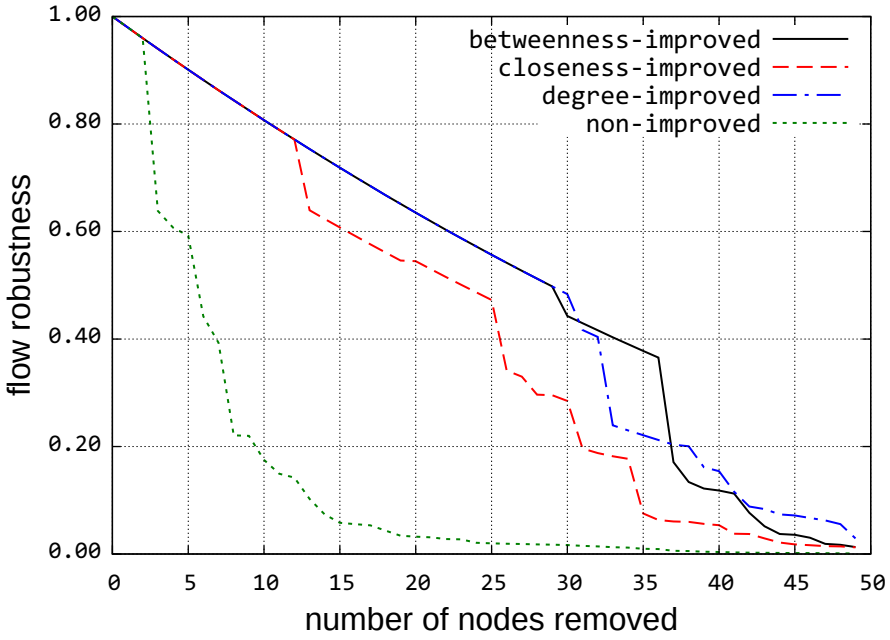

Figure 5.33: Level 3 degree-based attack

5.5 Spectral Metrics

In this section,we first study the accuracy of each spectra metric in predicting the three network resilience measures using a non-linear correlation. Then, we calculate and analyze the spectral robustness metrics for real-world networks. Third, we improve topology resilience of three real-world physical graphs via adding a set of links to maximize a given spectral metric.

5.5.1 Spectral Metrics Evaluation

We present our evaluation results for baseline and random graphs. We demonstrate how each robustness metric is related to the resilience of a given graph against centrality-based attacks. We use the three resilience graph measures: sums of flow robustness against degree attack (SFRD), sums of flow robustness against closeness attack (SFRC),

and sums of flow robustness against betweenness attack (SFRB), which are presented in Section 3.5.

Table 5.4: Baseline graphs robustness evaluation

Graph	Nodes	Links	λ_2	$\Delta\lambda$	$\hat{\tau}$	WS	$\bar{\lambda}$	SFRD	SFRC	SFRB
Full-mesh	10	45	10.00	10.00	0.20	1.00	9.66	3.67	3.67	3.67
Torus	9	18	3.00	3.00	0.50	1.27	2.87	3.14	3.14	3.14
Wheel	10	18	1.47	2.63	0.69	1.48	2.95	2.91	2.73	2.73
Grid	9	12	1.00	1.41	0.96	2.44	1.67	2.72	2.61	2.61
Ladder	10	13	0.38	0.73	1.25	3.04	1.61	2.62	2.47	2.47
Ring	10	10	0.38	0.38	1.83	3.75	1.19	2.56	2.29	2.29
Barbell	12	17	0.09	0.01	3.03	3.02	2.19	1.97	1.86	1.86
Linear	10	9	0.10	0.24	3.67	4.37	1.09	2.11	1.67	1.67
Tree	15	14	0.10	0.29	3.50	5.46	1.18	1.61	1.94	1.61
Star	10	9	1.00	3.00	1.80	2.00	1.49	1.00	1.00	1.00
corr(X, SFRD)			0.40	0.24	−0.67	−0.31	0.48			
corr(X, SFRC)			0.40	0.24	−0.67	−0.31	0.48			
corr(X, SFRB)			0.40	0.24	−0.67	−0.31	0.48			

Baseline graphs

We calculate spectral graph robustness metrics for each baseline graph and the results are shown in Table 5.4. We attempt to have 10 nodes in each graph unless it is structurally not possible e.g. grid, barbell, and tree graphs. The full-mesh graph is the most robust since it is fully connected. For this reason, the full-mesh graph has the highest values for the metrics: λ_2, $\Delta\lambda$, and $\bar{\lambda}$. Similarly, against centrality-based attacks, the mesh graph yields the highest robustness measures: SFRD, SFRC, and SFRB. The full-mesh graph yields the lowest (most robust) values for the metrics: $\hat{\tau}$ and WS. On the other hand, the star graph is one of the least robust graphs due to its single-node failure that causes complete disconnectivity. Against centrality-based attacks, the star graph yields the lowest value of 1 for all robustness measures because the flow robustness values are all zero after removing the first node. However, none of the spectral graph metrics capture

this vulnerability, as shown in Table 5.4. For example, the metrics λ_2, $\Delta\lambda$, $\hat{\tau}$, and $\bar{\lambda}$ values indicate that the 10-node star is more robust than 10-node ladder, which is clearly not shown by the resilience measures: SFRD, SFRC, and SFRB.

For non-linear comparison, we use Spearman's rank correlation non-linear coefficient, which yields 1 for perfect correlation, -1 for perfect inverse non-linear correlation, and 0 for no correlation [106]. We choose to use non-linear correlation because we want to capture the relation between graph metric values and number of connections is not linear and our objective is to measure how graph robustness values relate to our three resilience measurement values: SFRD, SFRC, and SFRB. Between the most resilient baseline (full-mesh) and the least resilient baseline (star) graph, we show seven other graphs with their spectral robustness metrics in Table 5.4. To determine the accuracy of each spectral metric in predicting the graph resilience against the three centrality-based attacks, we use three Spearman's rank-based correlation functions: corr(X, SFRD), corr(X, SFRC), and corr(X, SFRB), which are shown in Table 5.4. For example, the rank correlation value between λ_2 and the robustness measure SFRD is corr(λ_2, SFRD)= 0.40. By comparing all the correlation values, we observe that network criticality $\hat{\tau}$ consistently yields the best correlation values for the three correlation functions with corr($\hat{\tau}$, SFRD)= -0.67, corr($\hat{\tau}$, SFRC)= -0.67, and corr($\hat{\tau}$, SFRB)= -0.67. The minus sign indicates that the relation is inverse. For the baseline graphs, the second best robustness metric predictor is the natural connectivity robustness metrics with corr($\bar{\lambda}$, SFRD)= 0.48, corr($\bar{\lambda}$, SFRC)= 0.48, and corr($\bar{\lambda}$, SFRB)= 0.48. Relatively, the worst robustness metric predictor is the spectral gap with corr($\Delta\lambda$, SFRD)= 0.24, corr($\Delta\lambda$, SFRC)= 0.24, and corr($\Delta\lambda$, SFRB)= 0.24.

Random graphs

He calculate spectral graph robustness metrics for each random graph. We generate $10,000$ random graphs using 20 nodes for each graph type. We select this sample size

to achieve high significance level, i.e. p-value $\leq 10^{-3}$. We use a *non-linear* correlation function to see how each robustness metric is related to flow robustness metric. For each graph set, we record their spectral graph metrics and then apply the three centrality-based attacks. Next, we show the correlation rank value between the recorded spectral robustness values and the resilience measures: SFRD, SFRC, and SFRB. The correlation results are shown in Table 5.5.

For Gilbert random graphs, we select two alternatives with $p = 0.8$ and $p = 0.5$ to generate dense and semi-dense graphs. We observe that the correlation values are mostly smaller than other random graph types. This is because Gilbert graphs are completely random. Moreover, there is no consistency for highest correlation values among robustness metrics to predict network resilience, which tells us that none of the robustness metrics can be used as robustness measure for Gilbert random graphs.

For Waxman random graphs, we select three combinations of parameters: $(\alpha = 0.5, \beta = 0.5)$, $(\alpha = 0.5, \beta = 0.8)$, and $(\alpha = 0.5, \beta = 0.8)$. For the graphs with $\alpha = 0.5$ and $\beta = 0.5$, we obtain both medium density graphs and a medium number of long links. For the graphs with $\alpha = 0.5$ and $\beta = 0.8$, we get medium density graphs and a high number of long links. For the graphs with $\alpha = 0.8$ and $\beta = 0.5$, we obtain high density graphs and a medium number of long links. The maximum distance threshold L is set to 1 and the locations are randomly selected using a uniform distribution with a range of $[0, 1]$ for both x- and y-axis.

By observing all the correlation values for each Waxman graph combination, we see that network criticality $\hat{\tau}$ consistently yields the best correlation values for the three correlation functions. For example, for the resilience against betweenness attack (SFRB), the correlation values are between -0.85 and -0.78, which indicates a high inverse correlation. This means that most of the time, low network criticality values show high robustness.

Both algebraic connectivity and weighted spectrum correlation values are good predictors for network resilience, but they are inconsistent against three centrality-based attacks.

For Gabriel random graphs, network criticality $\hat{\tau}$ also yields consistently the best correlation values against all three centrality-based attacks with correlation values between -0.77 and -0.66. Without consistency, algebraic connectivity and weighted spectrum show the second and third best network resilience predictors.

Table 5.5: Random graphs robustness correlation values

Random Graph	SFRD					SFRC					SFRB				
	λ_2	$\Delta\lambda$	$\hat{\tau}$	WS	$\bar{\lambda}$	λ_2	$\Delta\lambda$	$\hat{\tau}$	WS	$\bar{\lambda}$	λ_2	$\Delta\lambda$	$\hat{\tau}$	WS	$\bar{\lambda}$
Gilbert $p = 0.8$	0.47	0.38	-0.47	-0.49	0.41	0.48	0.39	-0.47	-0.49	0.41	0.76	0.37	-0.49	-0.47	0.39
Gilbert $p = 0.5$	0.53	0.39	-0.62	-0.53	0.47	0.52	0.40	-0.59	-0.50	0.45	0.69	0.34	-0.61	-0.47	0.41
Waxman ($\alpha = 0.5, \beta = 0.5$)	0.63	0.43	-0.79	-0.66	0.60	0.65	0.45	-0.78	-0.68	0.60	0.74	0.44	-0.85	-0.72	0.60
Waxman ($\alpha = 0.5, \beta = 0.8$)	0.63	0.42	-0.76	-0.60	0.55	0.63	0.45	-0.74	-0.60	0.54	0.74	0.41	-0.81	-0.62	0.52
Waxman ($\alpha = 0.8, \beta = 0.5$)	0.61	0.41	-0.73	-0.58	0.52	0.59	0.45	-0.71	-0.58	0.52	0.76	0.39	-0.78	-0.58	0.48
Gabriel	0.53	0.16	-0.66	-0.60	0.57	0.65	0.28	-0.71	-0.57	0.52	0.72	0.27	-0.76	-0.61	0.52

5.5.2 Spectral Metrics Improvement

Here, we improve the spectral robustness of three physical networks by adding a set of links to maximize a given spectral graph metric. Then, we apply our algorithm to three physical networks using the five robustness functions while measuring their resilience.

Real-World Networks Improvement

In this section, we apply our algorithm presented in 4.4.1 to improve three real-world networks. The objective functions are set to either: maximize algebraic connectivity λ_2, minimize network criticality $\hat{\tau}$, maximize natural connectivity $\bar{\lambda}$, maximize spectral gap $\Delta\lambda$, or minimize weighted spectrum WS. In this research, we set the number of required links L_i to 30 for all input graphs.

For the three graphs, we apply the three centrality-based attacks on non- and improved graphs to evaluate the spectral robustness improved graph resilience against such attacks. While removing each node, we examine the flow robustness values. For Level 3, the results of applying betweenness-based, closeness-based, degree-based attacks are shown in Figure 5.34, Figure 5.35, and Figure 5.36. Moreover, we present sums of flow robustness during each attack and the spectral robustness values of non- and improved graph are shown in Table 5.6.

We observe that all improved graphs yield better results than non-improved graphs given that improved graphs have 30 additional links. Among the improved graphs, we see that $\hat{\tau}$-improved, i.e. network criticality improved, perform better than the other spectral metrics improved graphs. Among all nine scenarios, seven scenarios show $\hat{\tau}$-improved graphs have the highest resilience measures: SFRD, SFRC, and SFRB. The second best performing spectral robustness metric is algebraic connectivity, which shows the second highest resilience measures in the same seven scenarios. Moreover, λ_2-improved graphs have the highest resilience measures in the two remaining scenarios. The other spectral robustness metrics do not show consistent network resilience results. For example, the weighted spectrum improving approach yields mostly the worst resilience for CORONET graphs; however, the weighted spectrum improving approach yields the third best resilience values for the Level 3 graph. This inconsistency can be also seen for the other two metrics: spectral gap and natural connectivity. In this section, we have presented relevant plots to evaluate improved Level 3 and CORONET physical networks. The complete set of plots for evaluating other physical-level topologies is presented in Appendix B.3.

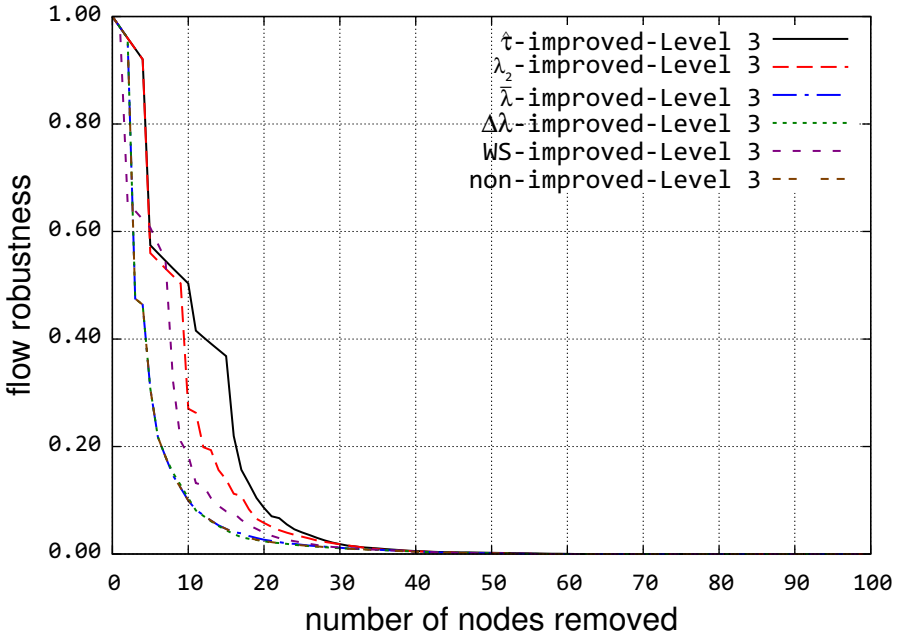

Figure 5.34: Level 3 betweenness-based attack

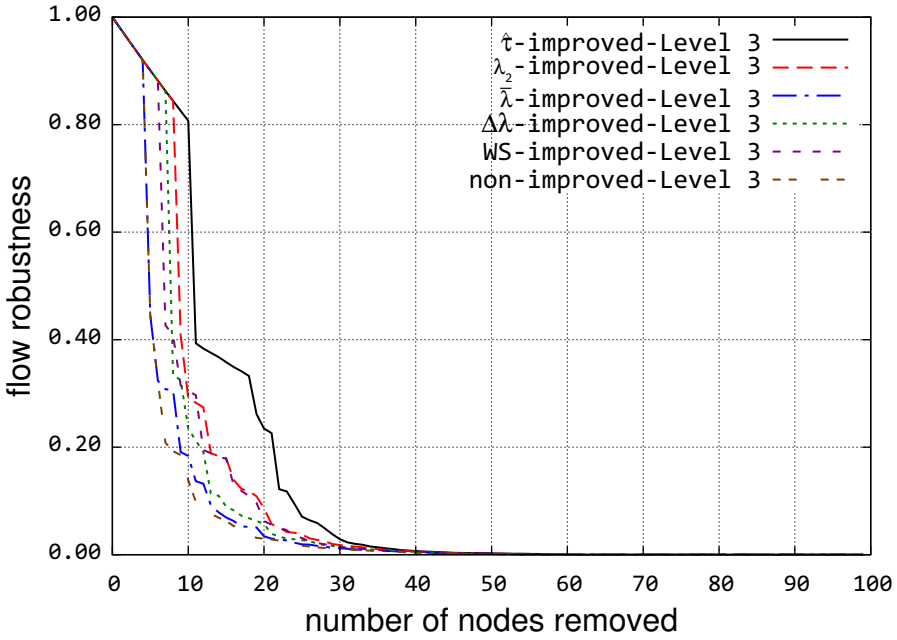

Figure 5.35: Level 3 closeness-based attack

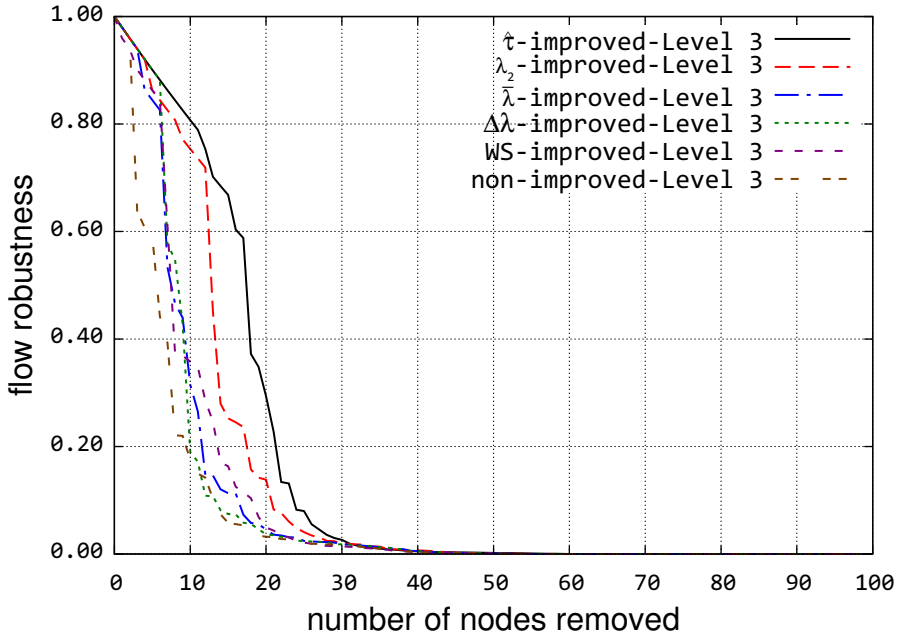

Figure 5.36: Level 3 degree-based attack

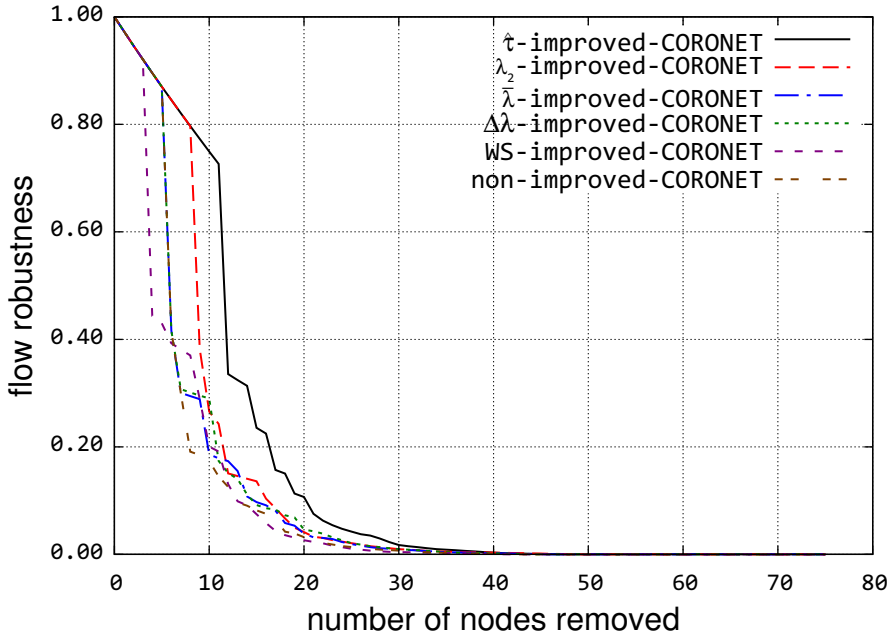

Figure 5.37: CORONET closeness-based attack

5.6 Comprehensive Evaluation of Metrics Accuracy

In this section, we compare the accuracy of several graph properties and graph robustness metrics to predict network resilience against targeted attacks. Using a set of baseline

Table 5.6: Real-world networks robustness evaluation

Graph	Nodes	Links	λ_2	$\Delta\lambda$	$\hat{\tau}$	WS	$\bar{\lambda}$	SFRD	SFRC	SFRB
non-improved-CORONET	75	99	0.04	0.27	2.52	18.11	1.59	9.87	9.87	7.41
λ_2-improved-CORONET	75	129	0.17	1.19	1.32	12.33	2.43	12.91	12.91	9.33
$\hat{\tau}$-improved-CORONET	75	129	0.14	0.48	1.09	11.18	2.02	14.23	14.23	9.36
$\bar{\lambda}$-improved-CORONET	75	129	0.04	3.99	2.22	15.61	5.09	12.53	12.53	6.76
$\Delta\lambda$-improved-CORONET	75	129	0.04	4.98	2.27	15.94	5.39	11.97	11.97	7.50
WS-improved-CORONET	75	129	0.08	0.14	1.33	9.85	2.24	10.44	10.44	7.03
non-improved-Internet2	57	65	0.04	0.16	3.40	16.84	1.36	4.71	4.71	4.05
λ_2-improved-Internet2	57	95	0.17	0.65	1.33	9.82	2.14	11.82	11.82	7.71
$\hat{\tau}$-improved-Internet2	57	95	0.16	0.46	1.12	8.68	1.96	10.43	10.43	8.20
$\bar{\lambda}$-improved-Internet2	57	95	0.04	5.77	3.03	14.50	6.06	7.26	7.26	3.97
$\Delta\lambda$-improved-Internet2	57	95	0.04	5.14	2.87	14.14	5.37	6.76	6.76	4.05
WS-improved-Internet2	57	95	0.08	0.13	1.49	7.47	2.27	5.78	5.78	4.46
non-improved-Level 3	99	132	0.03	0.15	3.24	23.80	1.71	7.30	7.30	5.68
λ_2-improved-Level 3	99	162	0.15	0.56	1.43	16.70	2.15	13.62	13.62	9.55
$\hat{\tau}$-improved-Level 3	99	162	0.12	0.35	1.23	15.77	2.02	16.72	16.72	11.24
$\bar{\lambda}$-improved-Level 3	99	162	0.03	5.39	2.97	22.00	5.94	9.71	9.71	5.71
$\Delta\lambda$-improved-Level 3	99	162	0.03	5.40	2.74	22.13	5.73	9.53	9.53	5.68
WS-improved-Level 3	99	162	0.05	0.32	1.54	14.33	2.11	10.12	10.12	7.47

graphs, we calculate graph properties and robustness metrics for each graph to give an intuition for how each metric is determined. We use three resilience graph measures: sums of flow robustness against degree attack (SFRD), sums of flow robustness against closeness attack (SFRC), and sums of flow robustness against betweenness attack (SFRB), which are presented in Section 3.5.

5.6.1 Baseline Graphs

In this section, we generate 10 nodes in each graph presented in Section 3.1.1 unless it is structurally impossible, e.g. grid, torus, barbell, and tree graphs, in which case we generate the feasible network closest to 10 nodes. Then, we apply the graph metrics presented in Section 2.2 and calculate their values for each generated baseline graph. Moreover, we measure the resilience of each graph against node attacks using the three

measures SFRD, SFRC, and SFRB, shown in Table 5.7. For each metric, we measure its accuracy in predicting the resilience of the graphs by correlation of its values with resilience measures. For example, the *accuracy* of the metric number of links $|L|$ in *predicting* the resilience measure SFRD is 0.74. By observing all the correlation values: corr(X, SFRD), corr(X, SFRC), and corr(X, SFRB), we notice that the total graph diversity TGD graph metric has the highest accuracy values of 0.99, 0.96, 0.96 for corr(X, SFRD), corr(X, SFRC), and corr(X, SFRB). Next, we observe that node average degree comes second with 0.91, 0.88, and 0.84 for the resilience measures. The third highest is variance of node-betweenness metric with accuracy values corr($\sigma^2_{C_{Bv}}$, SFRD | SFRC | SFRB) ≤ -0.85. Here the negative sign denotes an inverse correlation with robustness.

Among the presented graphs, the full-mesh graph has obviously the highest resilient since there is a link between every pair. On the other hand, the star graph has the lowest resilience because removing one node can fully disconnect the network. We observe that just only TGD and average node degree graph metric captures this fact by ranking the full-mesh as the highest and the star as the lowest. Although algebraic connectivity and network criticality do not rank the mesh and star graphs correctly, their over all accuracy in predicting the resilience is higher, corr(λ_2, SFRD | SFRC | SFRB) ≥ 0.71 and corr($\hat{\tau}$, SFRD | SFRC | SFRB) ≤ -0.84, than the rest (except TGD and average node degree).

5.6.2 Random Graphs

In the baseline graphs evaluation, we have studied and compared 10 structurally different graphs to give some intuition about robustness metrics. However, with just 10 graphs, we can not draw a solid conclusion about the accuracy of the compared metrics to predict their resilience against node attacks. In this section, we increase our graph sample size from 10 graphs to 30,000 graphs divided into six 5000-sample-size classes. We select this sample size to achieve high significance level, i.e. p-value $\leq 10^{-4}$. We note that all

Table 5.7: Evaluating graph robustness metrics using baseline graphs

	Star	Tree	Linear	Barbell	Ring	Ladder	Grid	Wheel	Torus	Mesh	corr(X,SFRD)	corr(X,SFRC)	corr(X,SFRB)
$\|N\|$	10.00	15.00	10.00	12.00	10.00	10.00	9.00	10.00	9.00	10.00	—	—	—
$\|L\|$	9.00	14.00	9.00	17.00	10.00	13.00	12.00	18.00	18.00	45.00	0.68	0.79	0.74
$\bar{C}_D$	1.80	1.87	1.80	2.83	2.00	2.60	2.67	3.60	4.00	9.00	0.84	0.88	0.91
$\sigma^2_{C_D}$	5.76	0.92	0.16	0.47	0.00	0.24	0.44	3.24	0.00	0.00	-0.58	-0.48	-0.55
$\sigma^2_{C_C}$	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	-0.43	-0.40	-0.44
$\sigma^2_{C_{B_v}}$	0.09	0.05	0.04	0.06	0.00	0.01	0.01	0.03	0.00	0.00	-0.88	-0.85	-0.85
$\sigma^2_{C_{B_l}}$	0.00	0.02	0.02	0.04	0.00	0.00	0.00	0.00	0.00	0.00	-0.56	-0.54	-0.54
CC	0.00	0.00	0.00	0.58	0.00	0.00	0.00	0.62	0.33	1.00	0.60	0.60	0.66
As	-1.00	-0.52	-0.12	0.13	1.00	0.28	-0.06	-0.33	1.00	1.00	0.66	0.61	0.68
R	1.00	3.00	5.00	4.00	5.00	3.00	2.00	1.00	2.00	1.00	-0.38	-0.46	-0.40
D	2.00	6.00	9.00	7.00	5.00	5.00	4.00	2.00	2.00	1.00	-0.60	-0.65	-0.62
$\bar{d}$	1.80	3.50	3.67	3.48	2.78	2.33	2.00	1.60	1.50	1.00	-0.71	-0.73	-0.73
TGD	0.00	0.00	0.00	0.23	0.39	0.68	0.73	0.82	0.91	1.00	0.96	0.96	0.99
λ_2	**1.00**	0.10	0.10	0.09	**0.38**	0.38	1.00	1.47	3.00	10.00	0.81	0.78	0.78
$\Delta\lambda$	3.00	0.29	0.24	0.01	0.38	0.73	1.41	2.63	3.00	10.00	0.58	0.60	0.57
$\hat{\tau}$	1.80	3.50	3.67	3.03	1.83	1.25	0.96	0.69	0.50	0.20	-0.84	-0.87	-0.87
WS	2.00	5.46	4.37	3.02	3.75	3.04	2.44	1.48	1.27	1.00	-0.67	-0.65	-0.71
$\bar{\lambda}$	1.49	1.18	1.09	2.19	1.19	1.61	1.67	2.95	2.87	9.66	0.75	0.77	0.82
C^*	0.11	0.04	0.05	0.06	0.11	0.16	0.23	0.29	0.44	1.00	0.87	0.84	0.88
SFRD	1.00	1.61	2.11	1.97	2.56	2.62	2.72	2.91	3.14	3.67	1.00	0.95	0.99
SFRC	1.00	1.94	1.67	1.86	2.29	2.47	2.61	2.73	3.14	3.67	0.95	1.00	0.96
SFRB	1.00	1.61	1.67	1.86	2.29	2.47	2.61	2.73	3.14	3.67	0.99	0.96	1.00
$\overline{\text{SFR}}$	**1.00**	1.72	1.82	1.90	**2.38**	2.52	2.64	2.79	3.14	3.67	0.99	0.96	1.00

randomly generated graphs are all connected to avoid zero values for spectral robustness metrics. The number of nodes in each generated graph is 20. Using three corr(X, SFRD), corr(X, SFRC), and corr(X, SFRB), we calculate the accuracy of each graph metric to predict resilience using a sample of 5000 graphs. The correlation results are shown in Table 5.8.

Gilbert graphs evaluation

In Table 5.8, the first two graph types are Gilbert random graphs with $p = 0.8$ and $p = 0.5$. The Gilbert random graphs are random graphs that do not model real-world communication networks. By observing the correlation values of the all metrics for the three attacks, we see that all graph metrics have a low accuracy in predicating network resilience i.e. corr(X, SFRD | SFRC | SFRB) are mostly lower than 0.70 because of the randomness in generating Gilbert graphs. However, among the graph metrics, the

Table 5.8: Evaluating graph robustness metrics using random graphs

	|L|	$\bar{C}_D$	$\sigma^2_{C_D}$	$\sigma^2_{C_C}$	$\sigma^2_{C_{B_n}}$	$\sigma^2_{C_{B_l}}$	CC	As	R	D	$\bar{d}_{ij}$	TGD	λ_2	$\Delta\lambda$	$\hat{\tau}$	WS	$\bar{\lambda}$	C^*
corr(X, SFRD)																		
Gilbert p=0.8	0.44	0.44	-0.43	-0.33	-0.53	-0.47	0.36	0.38	0.21	0.00	-0.42	0.53	0.47	0.39	-0.45	-0.47	0.41	0.47
Gilbert p=0.5	0.54	0.54	-0.38	-0.27	-0.66	-0.53	0.31	0.40	0.00	-0.04	-0.54	0.47	0.53	0.39	-0.61	-0.52	0.47	0.61
W(0.5, 0.5)	0.75	0.75	-0.06	-0.35	-0.81	-0.69	0.25	0.30	-0.12	-0.33	-0.70	0.74	0.63	0.43	-0.79	-0.66	0.61	0.79
W(0.5, 0.8)	0.68	0.68	-0.24	-0.49	-0.78	-0.68	0.23	0.35	0.13	-0.24	-0.67	0.67	0.64	0.42	-0.76	-0.60	0.55	0.76
W(0.8, 0.5)	0.63	0.63	-0.25	-0.47	-0.75	-0.64	0.24	0.33	0.43	0.15	-0.64	0.60	0.61	0.39	-0.73	-0.57	0.51	0.73
Gabriel	0.65	0.65	0.12	-0.01	-0.49	-0.50	0.26	0.14	-0.12	-0.31	-0.55	0.70	0.53	0.15	-0.66	-0.60	0.56	0.66
corr(X, SFRC)																		
Gilbert p=0.8	0.45	0.45	-0.44	-0.33	-0.54	-0.47	0.37	0.39	0.21	0.00	-0.42	0.54	0.49	0.40	-0.45	-0.47	0.42	0.48
Gilbert p=0.5	0.52	0.52	-0.35	-0.25	-0.64	-0.52	0.27	0.37	0.00	-0.04	-0.52	0.44	0.51	0.40	-0.58	-0.50	0.45	0.58
W(0.5, 0.5)	0.73	0.73	-0.02	-0.35	-0.83	-0.73	0.20	0.20	-0.12	-0.33	-0.71	0.72	0.64	0.46	-0.78	-0.67	0.60	0.78
W(0.5, 0.8)	0.68	0.68	-0.20	-0.50	-0.78	-0.71	0.18	0.26	0.13	-0.24	-0.68	0.67	0.65	0.46	-0.76	-0.61	0.55	0.76
W(0.8, 0.5)	0.63	0.63	-0.20	-0.45	-0.75	-0.66	0.20	0.27	0.43	0.16	-0.64	0.58	0.60	0.44	-0.71	-0.57	0.51	0.71
Gabriel	0.61	0.61	0.17	0.02	-0.58	-0.65	0.17	0.12	-0.15	-0.36	-0.62	0.71	0.65	0.27	-0.71	-0.58	0.52	0.71
corr(X, SFRB)																		
Gilbert p=0.8	0.43	0.43	-0.60	-0.41	-0.61	-0.69	0.32	0.25	0.15	0.00	-0.43	0.59	0.75	0.37	-0.49	-0.46	0.39	0.49
Gilbert p=0.5	0.49	0.49	-0.43	-0.29	-0.62	-0.64	0.23	0.28	0.00	-0.09	-0.50	0.42	0.69	0.33	-0.60	-0.46	0.40	0.60
W(0.5, 0.5)	0.76	0.76	-0.03	-0.40	-0.84	-0.81	0.22	0.15	-0.16	-0.41	-0.77	0.81	0.74	0.45	-0.85	-0.72	0.60	0.85
W(0.5, 0.8)	0.67	0.67	-0.24	-0.56	-0.78	-0.79	0.18	0.20	0.11	-0.31	-0.71	0.74	0.75	0.42	-0.81	-0.62	0.52	0.81
W(0.8, 0.5)	0.62	0.62	-0.26	-0.54	-0.73	-0.78	0.19	0.16	0.42	0.11	-0.68	0.66	0.76	0.39	-0.78	-0.58	0.48	0.78
Gabriel	0.62	0.62	0.18	0.06	-0.53	-0.68	0.17	0.10	-0.22	-0.43	-0.69	0.73	0.73	0.27	-0.77	-0.61	0.51	0.77

$\sigma^2_{C_{B_v}}$ metric has slightly higher accuracy than the other metrics for degree and closeness attacks. Moreover, we observe that the algebraic connectivity λ_2 has the highest accuracy, corr(λ_2, SFRB)$\geq$ 0.66, in predicting graph resilience against betweenness attack (SFRB). On the other hand, we notice that both radius and diameter graph properties have consistently the lowest accuracy in predicting network resilience for the two Gilbert graphs.

Waxman graphs evaluation

The next three random graphs are generated using Waxman models, W(α, β) , with three combination of parameters: ($\alpha = 0.5, \beta = 0.5$), ($\alpha = 0.5, \beta = 0.8$), and ($\alpha = 0.5, \beta = 0.8$). Waxman graphs exhibit mesh-like properties that can model logical-level networks with some long links to reduce diameter. For the graphs with $\alpha = 0.5$ and $\beta = 0.5$, we get obtain medium density graphs and medium number of long links. For the graphs with $\alpha = 0.5$ and $\beta = 0.8$, we get medium density graphs and a high number of long links.

For the graphs with $\alpha = 0.8$ and $\beta = 0.5$, we get high density graphs and a medium number of long links. The maximum distance threshold L is set to 1 and the locations are randomly selected using a uniform distribution with a range of $[0, 1]$ for both x-axis and y-axis.

Unlike Gilbert graphs results, some Waxman graph metrics have high accuracy values in measuring resilience. We observe that the variance of node-betweenness metric has a slightly higher accuracy than the other metrics for degree and closeness attacks, i.e. $\text{corr}(\sigma^2_{\text{C}_{\text{B}_\text{n}}}, \text{SFRD} \mid \text{SFRC}) \leq -0.75$. Next, both network criticality $\hat{\tau}$ and effective graph resistance C^* are the second best predictors for graph resilience against both degree and closeness attacks. In fact, both metrics have *almost* identical accuracy results because they both claim to measure *graph resistance* using the eigenvalues of the Laplacian matrix.

For the betweenness attack, both network criticality $\hat{\tau}$ and effective graph resistance C^* are the best predictor for graph resilience with $\text{corr}(\hat{\tau}, \text{SFRB}) \leq -0.78$. We also note that both $\sigma^2_{\text{C}_{\text{B}_\text{l}}}$ and TGD metrics have the second- and third-best results for the betweenness attack. Moreover, we observe that the radius graph property *generally* performs very poorly in predicting the graph resilience.

Gabriel graphs evaluation

The sixth random graph class is generated using Gabriel graphs that exhibit mesh-like structure and model physical-level networks. By observing the correlation values of the all metrics for the three attacks, we see that the total graph diversity TGD has the best accuracy values for predicting network resilience against both degree and closeness attacks with $\text{corr}(\text{TGD}, \text{SFRD} \mid \text{SFRC}) \geq 0.70$. For the betweenness attack, we see that both $\hat{\tau}$ and C^* are the best predictors for graph resilience.

5.7 Comprehensive Evaluation of Improved Graphs

In this section, we conduct a comprehensive evaluation of improved graph resilience of both un- and weighted graphs. We start applying our algorithm presented in Section 4.4.1 to improve the three real-world graphs presented in Section 3.1.3.

5.7.1 Unweighted Real-World Improved Graphs

We apply our algorithm presented in Section 4.4.1 to three unweighted physical graphs: CORONET, Internet2, and Level 3, which are discussed in Section 3.1.3. Then we apply three centrality-based attacks while measuring network resilience against such attacks. Table 5.9 shows a summary our evaluation of each improved graph of the three physical networks.

For Level 3, the results of applying betweenness-, closeness-, and degree-based attacks are shown in Figure 5.38, Figure 5.39, and Figure 5.40 respectively. The Level 3-$\sigma^2_{C_{B_l}}$-improved graph indicates consistent *best* results against the three centrality-based attacks with 14.62, 13.75, and 10.37 for the degree-based, closeness-based, and betweenness-based attacks. Hence, adding links to the Level 3 physical network to increase link-betweenness yields the best outcome in terms of network resilience against centrality-based attacks. Next, the Level 3-$\sigma^2_{C_{B_n}}$-improved graph also shows consistent *second best* network resilience results. On the other hand, Level 3-CC-improved graphs show the *worst* results among all the other graph robustness improvement approaches with 7.80, 7.68, and 5.70 for the degree-, closeness-, and betweenness-based attacks. In this section, we have presented relevant plots to evaluate improved Level 3. The complete set of plots for evaluating other physical-level topologies is presented in Appendix B.4.

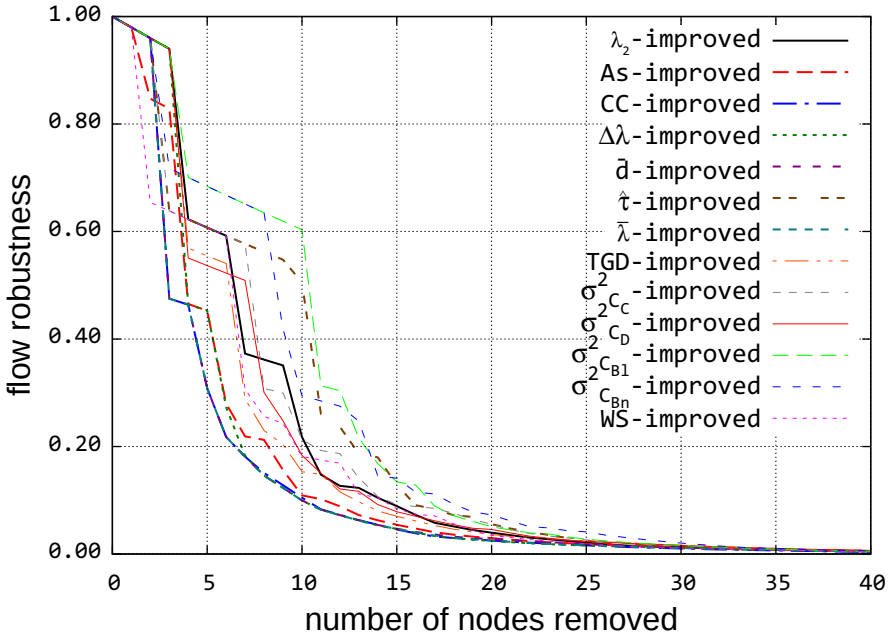

Figure 5.38: Level 3 betweenness attack

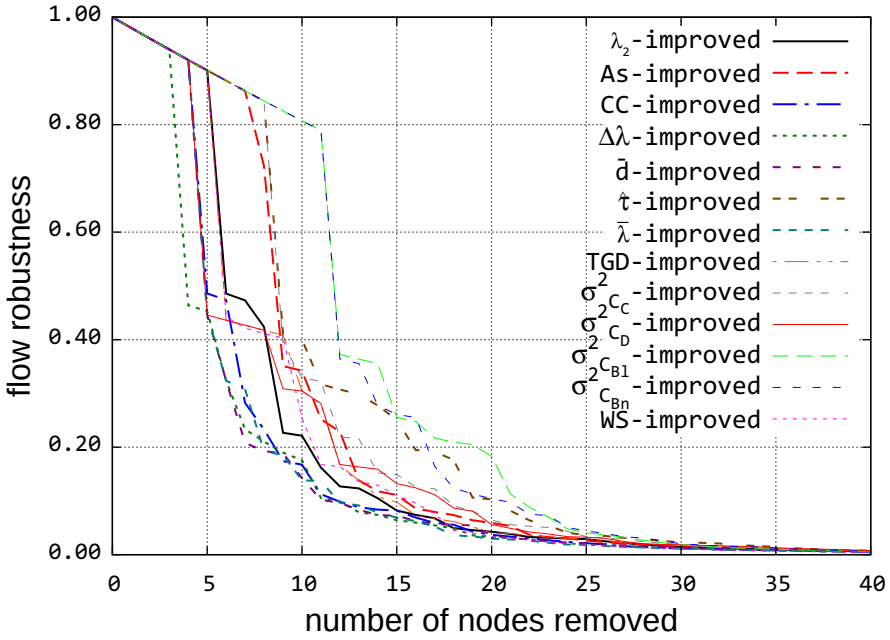

Figure 5.39: Level 3 closeness attack

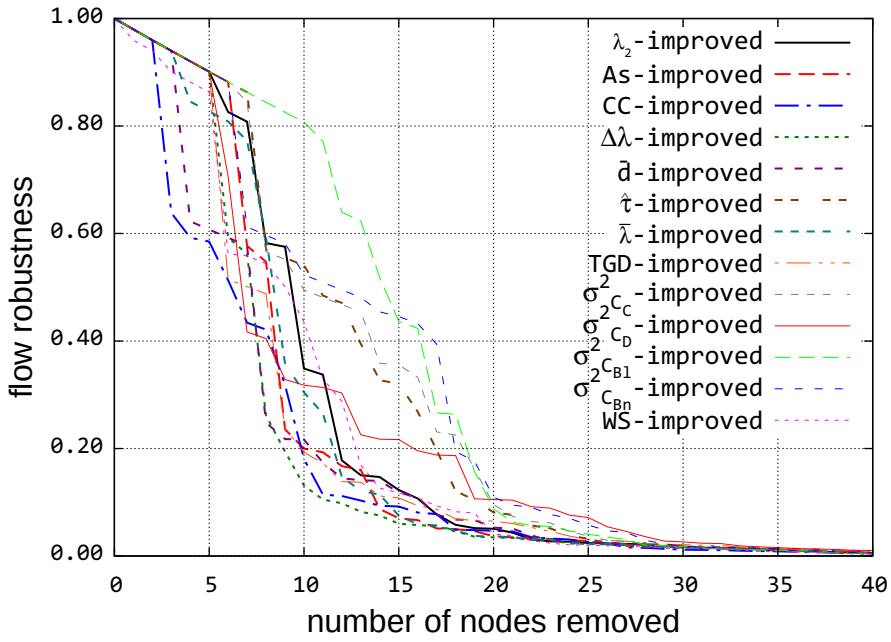

Figure 5.40: Level 3 degree attack

5.7.2 Evaluation of Improved Weighted Graphs

Here, we apply our algorithm presented in Section 4.4.1 to weighted graphs.

Weighted graphs via population-based model

For our dataset, we start with applying our population-based weighted graph model, which is discussed in Section 3.1.6, to convert three unweighted physical graphs: CORONET, Internet2, and Level 3. We note that we convert the resulting weights to logarithmic scale to avoid any overflow while computing graph robustness metrics. Then, we apply our algorithm presented in Section 4.4.1 to the weight graphs to generate improved graphs based on improving graph robustness metrics. To evaluate the improved graphs, we compute their weighted flow robustness while applying the three centrality attacks presented in Section 3.4. A summary of wighted flow robustness values is shown in Table 5.10.

For CORONET, the results of applying betweenness-, closeness-, and degree-based attacks are shown in Figure 5.41, Figure 5.42, and Figure 5.43. The CORONET-$\bar{d_{ij}}$-improved graph shows consistent best results against the three centrality-based attacks with sums of weighted flow robustness: 10430.15, 6514.62, 6024.30 for the degree-, closeness-, and betweenness-based attacks respectively. We observe that the assortativity improved graph for CORONET has the second highest sum of weight flow robustness for degree-based attack with 9381.73. However, assortativity improved graph is not consistent for other attacks. We also observe the CORONET-WS-improved consistently has the lowest sum weighted flow robustness values against centrality-based attacks. In fact, weighted spectrum improved graphs have the worst network resilient against centrality attacks as shown in Table 5.10. Here, we have presented relevant plots to evaluate improved CORONET. The complete set of plots for evaluating other physical-level topologies is presented in Appendix B.5. For Internet2, we observe that the Internet2-$\bar{d_{ij}}$-improved graph yields the highest resilience against centrality-based attacks with 7044.33, 5327.80, and 3559.27 for degree-, closeness-, and betweenness-based attacks. This indicates adding links to minimize the avaerge shortest path length, has the best network resilience outcome for CORONET and Internet2. However, for Level 3, minimizing the average shortest path length yields the second-best resilience against centrality-based attacks while adding links to balance node degree shows the best outcome.

Real-world weighted graphs

In this section, we apply our algorithm, presented in Section 4.4.1, to three real-world networks: GÉANT, CARNet, and InternetMCI, which are discussed in Section 3.1.4. To evaluate the improved graphs, we compute their weighted flow robustness while apply-

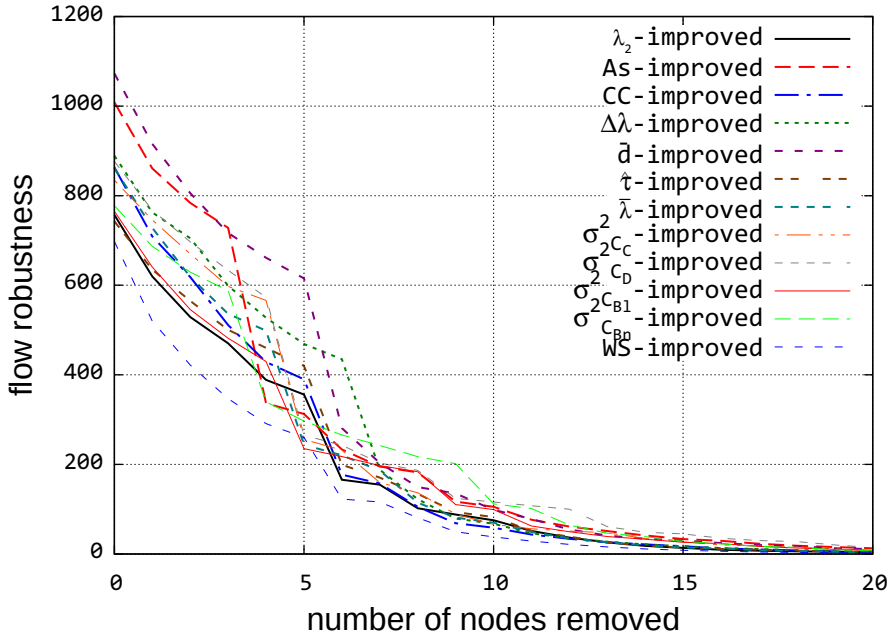

Figure 5.41: CORONET betweenness attack

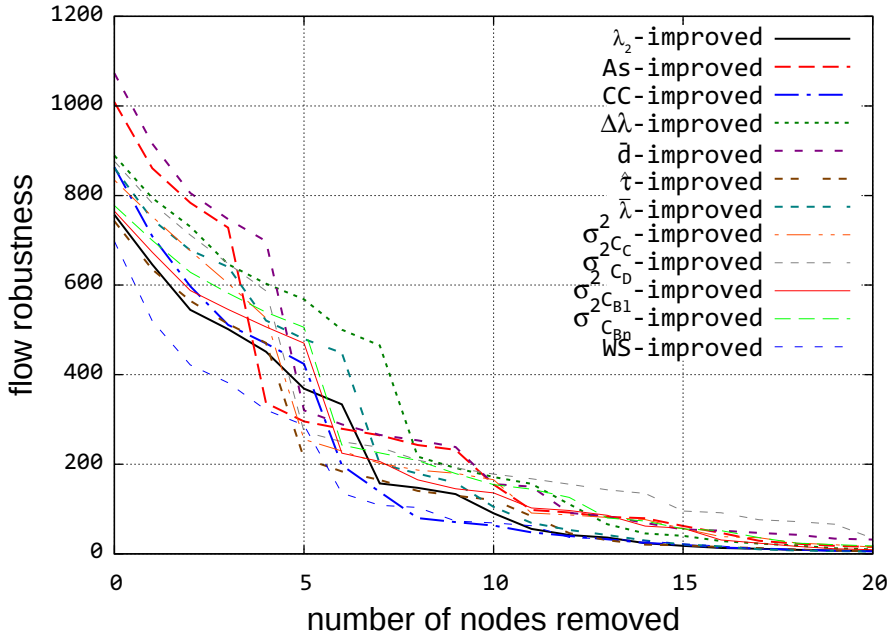

Figure 5.42: CORONET closeness attack

ing the three centrality attacks presented in Section 3.4. A summary of weighted flow robustness sum values is presented in Table 5.11.

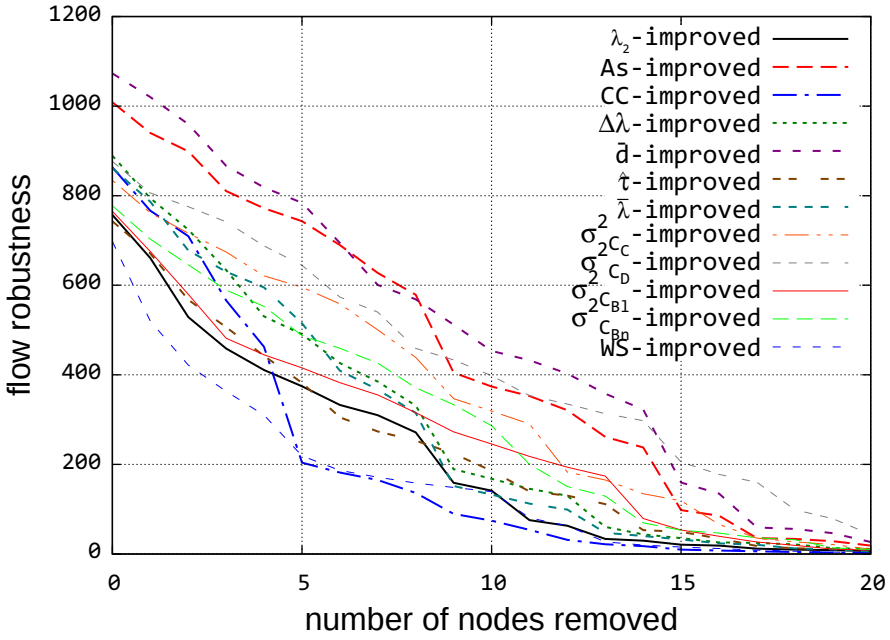

Figure 5.43: CORONET degree attack

For the GÉANT network, the results of applying betweenness-, closeness-, and degree-based attacks are shown in Figure 5.44, Figure 5.45, and Figure 5.46. The GÉANT-$\sigma^2_{C_D}$-improved graph shows the best weighted flow robustness sums with 3595.01, 3651.72, and 3407.27 for the degree-based, closeness-based, and betweenness-based attacks respectively. This indicates that adding links to balance node degree increases the network resilience against centrality-based attacks for GÉANT network. On the other hand, GÉANT-$\bar{\lambda}$-improved graph has the lowest weighted flow robustness sums with 1990.79, 1996.02, and 1815.88 for the degree-, closeness-, and betweenness-based attacks. This indicates that the natural connectivity improved graph has the worst network resilience outcome against centrality-based attacks. Here, we have presented relevant plots to evaluate improved GÉANT. The complete set of plots for evaluating other real-world physical-level topologies is presented in Appendix B.6. By studying the robustness values presented in Table 5.11, we observe that adding links to balance node-degree yields the best network resilience against centrality-based attacks for all studied networks: GÉANT,

CARNet, and InternetMCI.

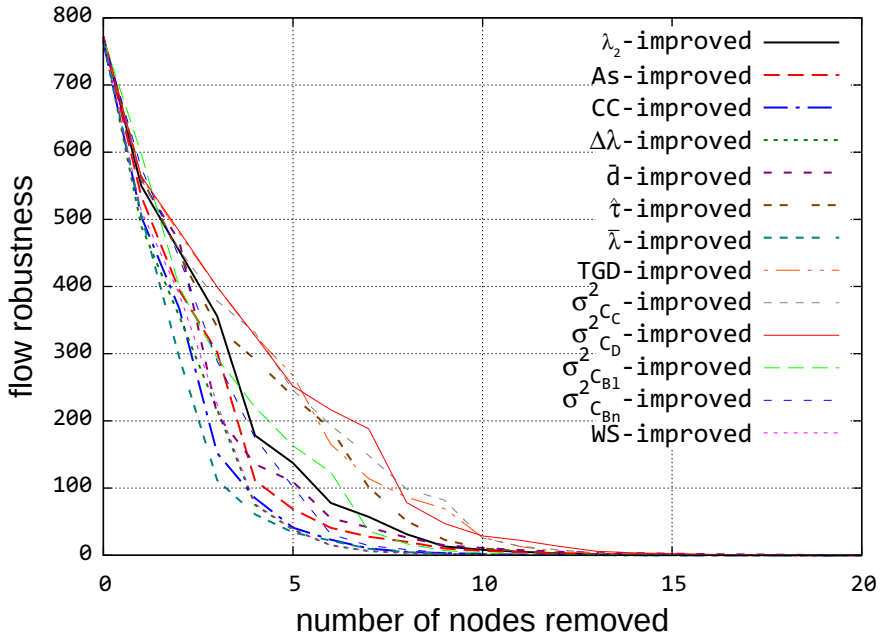

Figure 5.44: GÉANT betweenness attack

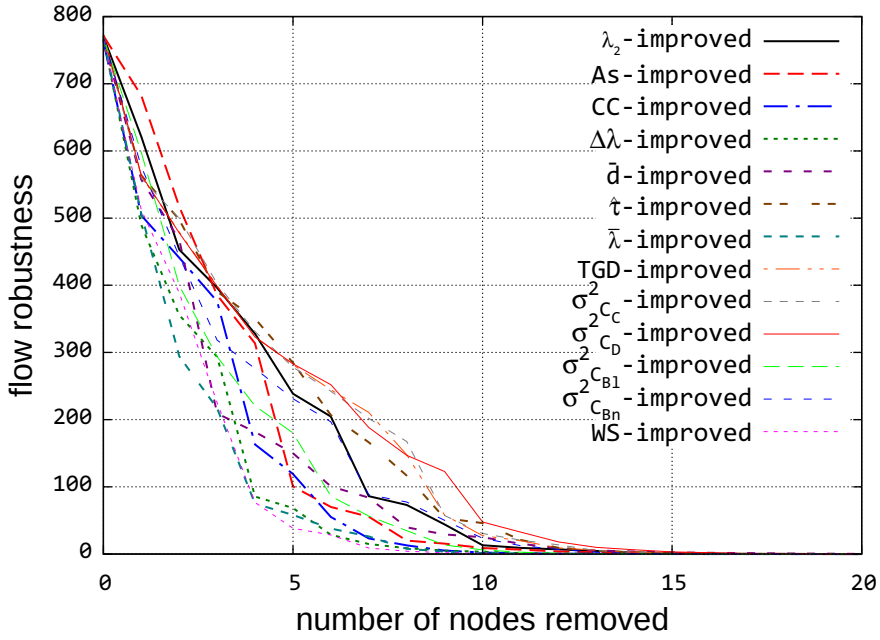

Figure 5.45: GÉANT closeness attack

Table 5.9: Evaluating robustness improvements for unweighted graphs

Physical Network	Degree-based	Closeness-based	Betweenness-based
CORONET-λ_2-improved	11.60	9.30	8.36
CORONET-As-improved	12.09	7.99	7.14
CORONET-CC-improved	10.02	7.80	6.56
CORONET-$\Delta\lambda$-improved	11.40	8.32	7.50
CORONET-$\bar{d_{ij}}$-improved	13.56	8.76	7.91
CORONET-$\hat{\tau}$-improved	13.07	10.28	8.20
CORONET-$\bar{\lambda}$-improved	11.67	8.01	7.52
CORONET-$\sigma^2_{C_C}$-improved	12.31	9.22	7.67
CORONET-$\sigma^2_{C_D}$-improved	14.40	8.20	7.58
CORONET-$\sigma^2_{C_{B_l}}$-improved	11.60	11.13	8.66
CORONET-$\sigma^2_{C_{B_n}}$-improved	13.55	11.55	8.57
CORONET-WS-improved	9.68	7.47	7.06
CORONET-TGD-improved	13.41	11.55	8.55
Internet2-λ_2-improved	8.90	7.26	7.29
Internet2-As-improved	9.09	5.78	5.26
Internet2-CC-improved	5.01	5.06	4.54
Internet2-$\Delta\lambda$-improved	7.37	5.56	4.03
Internet2-$\bar{d_{ij}}$-improved	7.83	6.31	4.28
Internet2-$\hat{\tau}$-improved	8.40	7.16	6.64
Internet2-$\bar{\lambda}$-improved	5.54	5.18	4.16
Internet2-$\sigma^2_{C_C}$-improved	7.16	6.03	5.66
Internet2-$\sigma^2_{C_D}$-improved	10.83	6.11	5.70
Internet2-$\sigma^2_{C_{B_l}}$-improved	8.98	7.21	6.58
Internet2-$\sigma^2_{C_{B_n}}$-improved	9.77	9.30	6.63
Internet2-WS-improved	4.67	5.69	4.75
Internet2-TGD-improved	10.12	7.33	5.92
Level 3-λ_2-improved	10.50	8.79	8.18
Level 3-As-improved	9.46	10.47	6.41
Level 3-CC-improved	7.80	7.68	5.70
Level 3-$\Delta\lambda$-improved	8.44	6.87	6.34
Level 3-$\bar{d_{ij}}$-improved	8.50	7.32	5.76
Level 3-$\hat{\tau}$-improved	12.35	11.93	9.32
Level 3-$\bar{\lambda}$-improved	9.73	7.36	5.68
Level 3-$\sigma^2_{C_C}$-improved	12.72	11.07	8.28
Level 3-$\sigma^2_{C_D}$-improved	10.77	8.99	7.97
Level 3-$\sigma^2_{C_{B_l}}$-improved	14.62	13.75	10.37
Level 3-$\sigma^2_{C_{B_n}}$-improved	13.28	13.38	9.79
Level 3-WS-improved	9.89	8.90	7.28
Level 3-TGD-improved	9.21	9.22	7.56

Table 5.10: Evaluating robustness improvements for population-based weighted graphs

Physical Network	Degree-based	Closeness-based	Betweenness-based
CORONET-λ_2-improved	4712.15	4376.82	3899.71
CORONET-As-improved	9381.73	5791.27	5281.90
CORONET-CC-improved	4388.05	4353.74	4285.38
CORONET-$\Delta\lambda$-improved	6102.39	6326.00	5054.87
CORONET-$\bar{d_{ij}}$-improved	10430.15	6514.62	6024.30
CORONET-$\hat{\tau}$-improved	5146.35	4148.58	4206.44
CORONET-$\bar{\lambda}$-improved	5864.78	5299.41	4356.62
CORONET-$\sigma^2_{C_C}$-improved	7457.33	5193.96	4674.90
CORONET-$\sigma^2_{C_D}$-improved	9094.29	6109.17	5230.00
CORONET-$\sigma^2_{C_{B_l}}$-improved	5786.06	4956.59	4212.64
CORONET-$\sigma^2_{C_{B_n}}$-improved	6406.11	5423.42	4737.15
CORONET-WS-improved	3586.61	3374.56	3062.07
Internet2-λ_2-improved	3844.34	3231.05	2974.61
Internet2-As-improved	6225.39	4104.42	3773.49
Internet2-CC-improved	2334.44	2731.45	2380.37
Internet2-$\Delta\lambda$-improved	4736.14	4757.46	3639.58
Internet2-$\bar{d_{ij}}$-improved	7044.33	5327.80	3559.27
Internet2-$\hat{\tau}$-improved	3099.54	2667.66	2561.42
Internet2-$\bar{\lambda}$-improved	4149.55	3207.58	3343.96
Internet2-$\sigma^2_{C_C}$-improved	4816.39	3916.03	3662.18
Internet2-$\sigma^2_{C_D}$-improved	4835.10	3992.52	3272.50
Internet2-$\sigma^2_{C_{B_l}}$-improved	3926.28	3747.95	3079.85
Internet2-$\sigma^2_{C_{B_n}}$-improved	4699.29	4081.54	3599.80
Internet2-WS-improved	2017.65	2384.85	1790.56
Level 3-λ_2-improved	5896.48	5122.69	4867.06
Level 3-As-improved	7394.54	8623.41	4827.91
Level 3-CC-improved	5192.12	5261.19	4023.00
Level 3-$\Delta\lambda$-improved	6719.40	6167.78	4540.16
Level 3-$\bar{d_{ij}}$-improved	8386.53	7174.46	5574.43
Level 3-$\hat{\tau}$-improved	6124.60	5623.22	5384.33
Level 3-$\bar{\lambda}$-improved	6797.31	6705.19	5120.55
Level 3-$\sigma^2_{C_C}$-improved	8232.40	7104.23	5486.29
Level 3-$\sigma^2_{C_D}$-improved	9444.25	7830.35	5656.45
Level 3-$\sigma^2_{C_{B_l}}$-improved	5727.69	5234.82	5073.34
Level 3-$\sigma^2_{C_{B_n}}$-improved	6220.91	6225.07	5762.63
Level 3-WS-improved	3563.84	3478.68	3180.70

Table 5.11: Evaluating robustness improvements for real-world weighted graphs

Graph	Degree-based	Closeness-based	Betweenness-based
GÉANT-λ_2-improved	2689.34	3254.40	2645.45
GÉANT-TGD-improved	3430.82	3547.47	3293.76
GÉANT-$\sigma^2_{C_C}$-improved	3485.94	3582.98	3316.64
GÉANT-CC-improved	2109.32	2467.93	1959.08
GÉANT-WS-improved	2088.80	2053.28	2036.48
GÉANT-$\bar{d_{ij}}$-improved	2840.92	2650.70	2420.92
GÉANT-$\sigma^2_{C_{B_n}}$-improved	2475.68	3074.53	2416.86
GÉANT-$\hat{\tau}$-improved	3112.70	3487.71	3037.95
GÉANT-$\sigma^2_{C_{B_l}}$-improved	2454.03	2662.36	2633.43
GÉANT-$\Delta\lambda$-improved	2035.75	2123.06	1971.23
GÉANT-As-improved	3025.59	2957.06	2299.24
GÉANT-$\sigma^2_{C_D}$-improved	**3595.01**	**3651.72**	**3407.27**
GÉANT-$\bar{\lambda}$-improved	1990.79	1996.02	1815.88
CARNet-λ_2-improved	785.55	783.64	783.86
CARNet-TGD-improved	884.20	885.12	882.91
CARNet-$\sigma^2_{C_C}$-improved	761.88	814.33	761.42
CARNet-CC-improved	535.88	535.88	535.88
CARNet-WS-improved	592.08	592.08	592.08
CARNet-$\bar{d_{ij}}$-improved	559.37	559.37	559.10
CARNet-$\sigma^2_{C_{B_n}}$-improved	683.13	806.41	683.13
CARNet-$\hat{\tau}$-improved	609.27	609.27	609.27
CARNet-$\sigma^2_{C_{B_l}}$-improved	638.77	671.50	632.91
CARNet-$\Delta\lambda$-improved	591.21	591.21	591.21
CARNet-As-improved	591.07	595.89	590.46
CARNet-$\sigma^2_{C_D}$-improved	**971.71**	**945.40**	**913.42**
CARNet-$\bar{\lambda}$-improved	610.69	610.69	610.69
InternetMCI-λ_2-improved	1331.28	1448.18	1344.51
InternetMCI-TGD-improved	1537.10	1589.06	1503.54
InternetMCI-$\sigma^2_{C_C}$-improved	1647.56	1777.40	1590.09
InternetMCI-CC-improved	1008.40	932.71	969.92
InternetMCI-WS-improved	999.50	1169.75	1005.81
InternetMCI-$\bar{d_{ij}}$-improved	1504.41	1465.11	1196.64
InternetMCI-$\sigma^2_{C_{B_n}}$-improved	1441.09	1526.83	1503.74
InternetMCI-$\hat{\tau}$-improved	1464.10	1434.94	1429.81
InternetMCI-$\sigma^2_{C_{B_l}}$-improved	1230.81	1204.31	1211.90
InternetMCI-$\Delta\lambda$-improved	1191.46	1101.51	1107.31
InternetMCI-As-improved	**1768.10**	1647.31	1426.78
InternetMCI-$\sigma^2_{C_D}$-improved	1731.64	**1656.70**	**1514.42**
InternetMCI-$\bar{\lambda}$-improved	1180.94	1136.56	1107.23

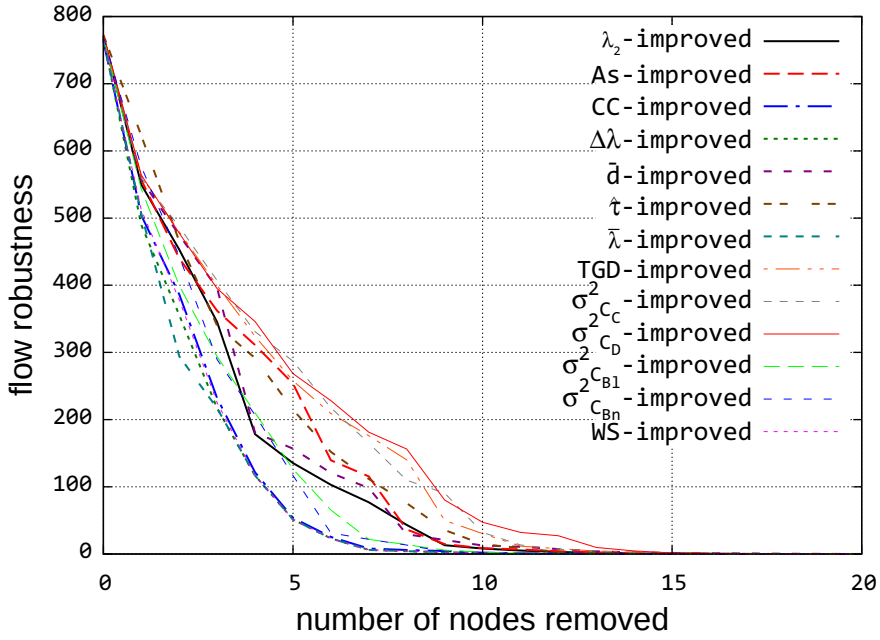

Figure 5.46: GÉANT closeness attack

5.8 Resilience State-Space Model Evaluation

We apply the resilience space-state model evaluation [7, 9, 87] on three networks: Level 3, Internet2, and CORONET, which are presented in Section 3.1.3. The resilience state model is discussed in Section 2.3. We define our operational state s as the ratio of operational nodes to the total number of nodes in the graph. On the other hand, we define our service model to be the end-to-end connectivity p between node pairs, which is the flow robustness of a given graph. For evaluation, we apply three centrality-based attacks presented in Section 3.4 to the improved graphs for each real-world network, presented in Section 5.7.2. The applied challenge is set to remove the 15% of the total number of nodes since after 15%, the network connectivity reaches the lowest point in the studied networks. Thus, we define our operational state as follows. Under normal conditions, the *normal state* is defined as 95% of nodes are operational. As challenges

impact the network, the *partially degraded* is defined to have 90%–95% operational nodes. As the network is severely impacted, the *severely degraded* is defined as the number is operational nodes is below 90%.

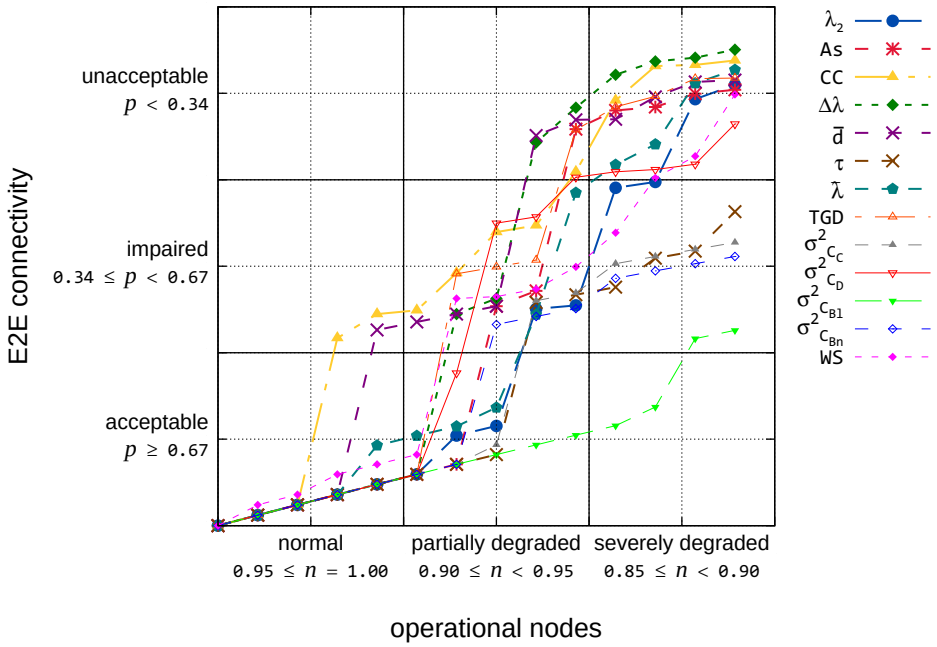

Figure 5.47: Resilience state space for Level 3 with degree attack

For Level 3 improved graphs, the results of applying degree-, closeness-, and betweenness-based attacks are shown in Figure 5.47, Figure 5.48, and Figure 5.49. Generally, the results match our improvement evaluation, discussed in Section 5.7.1, which shows the results of applying centrality-based attack to remove all nodes while in this evaluation, we only the remove 15% of the total number of graph. For the degree- and closeness-based attacks, we observe that the Level 3-$\sigma^2_{C_{B_1}}$-improved graph stays in *acceptable* service state for *normal* and *partially degraded* states. Then, it starts to move to the *impaired state* as the network enters the *severely degraded* state. For betweenness-based attack, the attack impact is more sever than degree- and closeness-based attacks since none of the networks provide an *acceptable* service with *partially degraded* operation. However, we still observe that the the Level 3-$\sigma^2_{C_{B_1}}$-improved graph outperform other improved graphs.

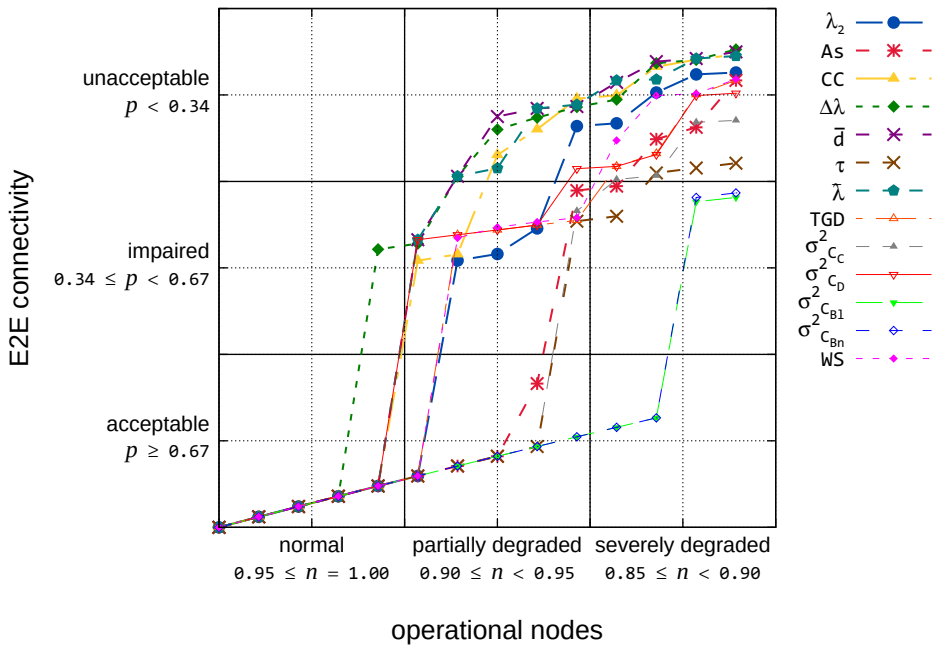

Figure 5.48: Resilience state space for Level 3 with closeness attack

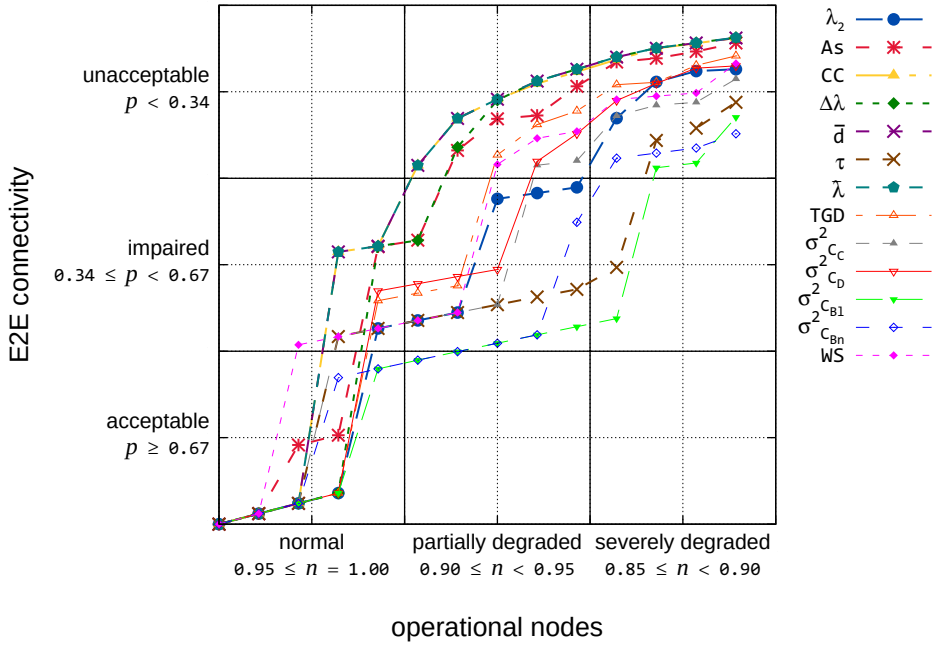

Figure 5.49: Resilience state space for Level 3 with betweenness attack

Hence, adding links to Level 3 network to balance link-betweenness has the best network resilience against centrality-based attacks.

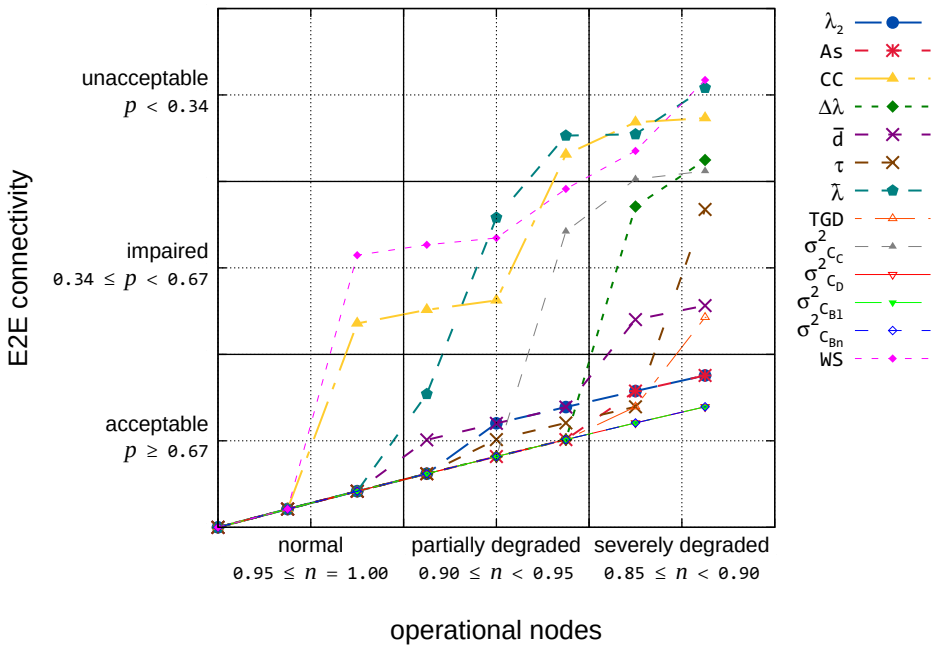

Figure 5.50: Resilience state space for Internet2 with degree attack

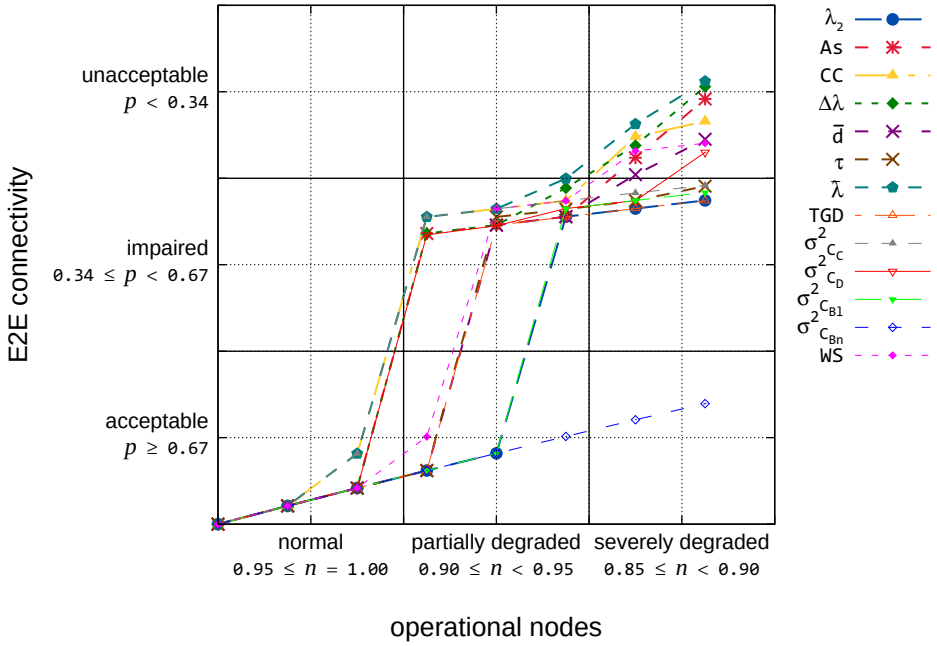

Figure 5.51: Resilience state space for Internet2 with closeness attack

For Internet2 improved graphs, the results of applying degree-, closeness-, and betweenness-based attacks are shown in Figure 5.50, Figure 5.51, and Figure 5.52. For the degree-

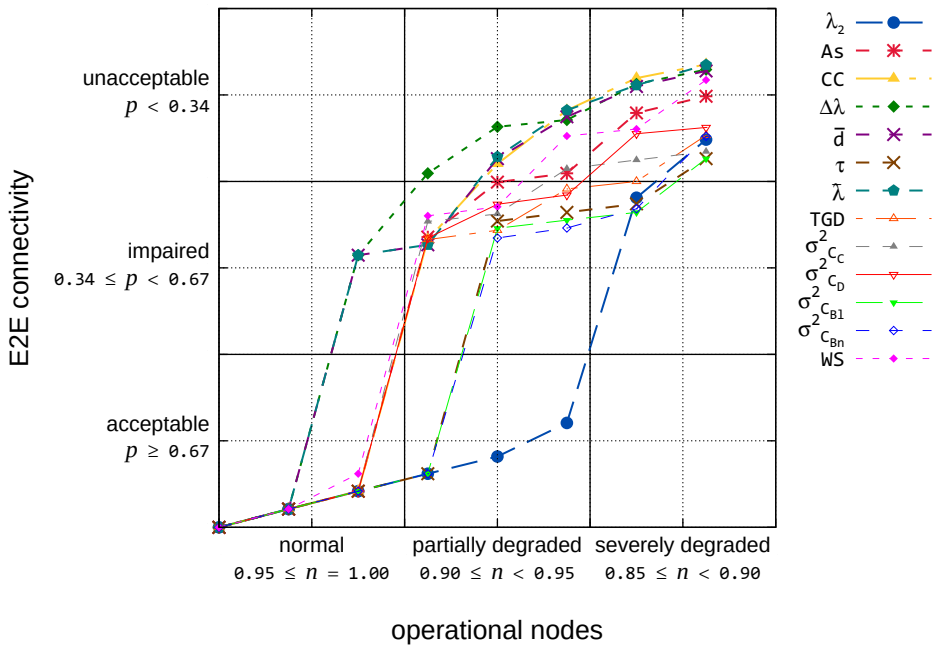

Figure 5.52: Resilience state space for Internet2 with betweenness attack

based attacks, we observe that several improved graphs such as Internet2-λ_2-improved, Internet2-$\sigma^2_{C_{B_1}}$-improved, and Internet2-$\hat{\tau}$-improved graphs stay in *acceptable* service state for *normal* and *partially degraded* states. However, only the Internet2-$\sigma^2_{C_{B_1}}$-improved graph keeps the same good behavior for closeness-based attack. For the betweenness-based attack, we observe that the Internet2-λ_2-improved graph outperforms the other improved graphs. These results indicate that for Internet2, both adding links to maximize algebraic connectivity or balance link-betweenness yeild the best results against centrality-based attacks.

For CORONET improved graphs, the results of applying degree-, closeness-, and betweenness-based attacks are shown in Figure 5.53, Figure 5.54, and Figure 5.55. For the degree-based attacks, we observe that most of the improved graphs stay in *acceptable* service state for *normal* and *partially degraded* states, except for CORONET-As-improved, CORONET-CC-improved, and CORONET-WS-improved graphs. This indicates that adding links

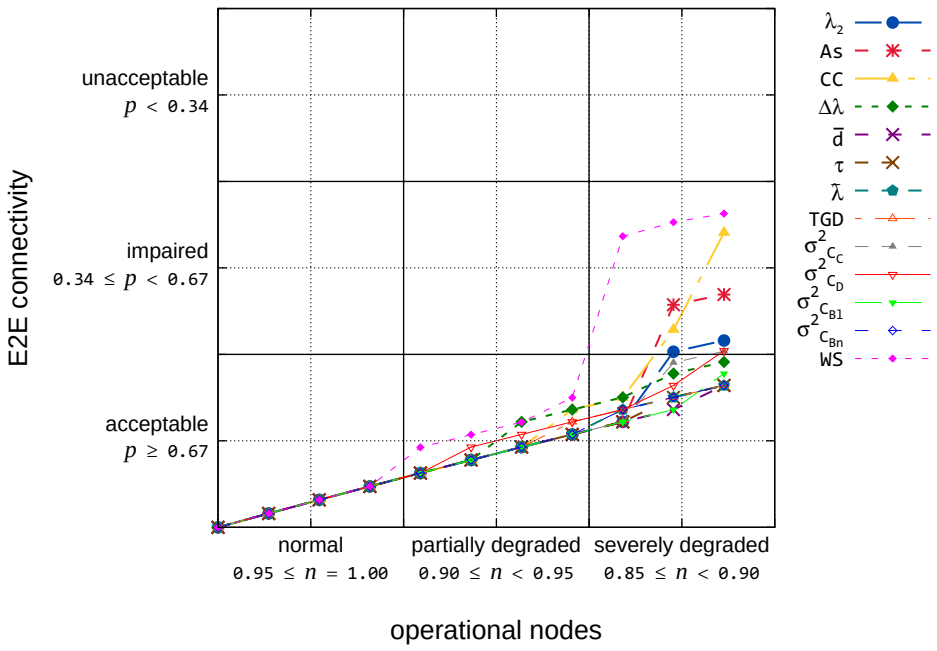

Figure 5.53: Resilience state space for CORONET with degree attack

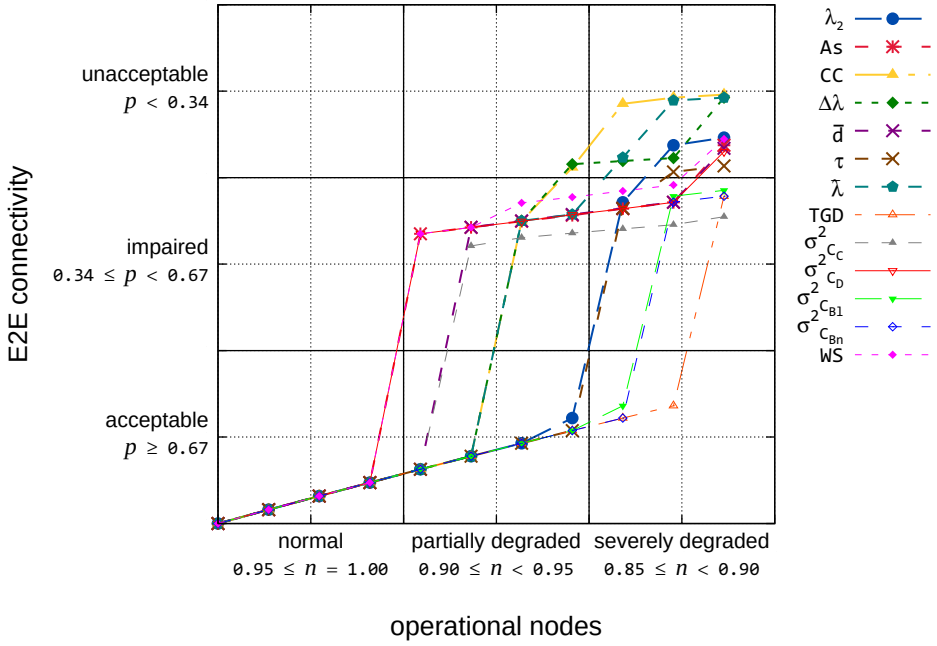

Figure 5.54: Resilience state space for CORONET with closeness attack

to maximize cluster coëficcient, for example, does not improve network resilience against centrality-based attacks. We observe that the CORONET-$\sigma^2_{C_{B_1}}$-improved graph con-

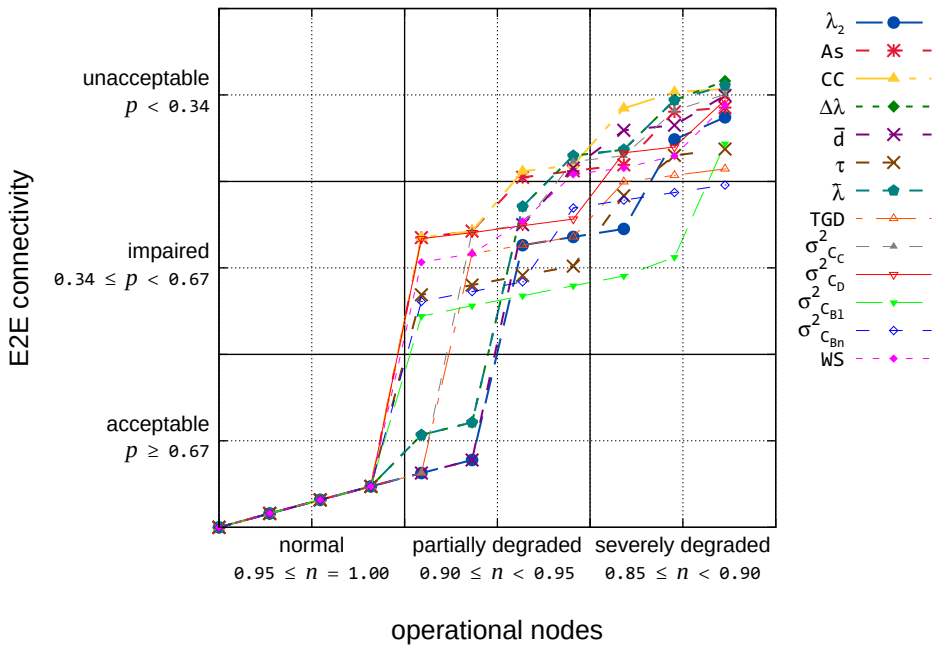

Figure 5.55: Resilience state space for CORONET with betweenness attack

sistently yields the best network resilience against such attacks. Thus, adding links to balance link-betweenness produces the highest network resilience against centrality-based attacks.

By studying the results for all networks presented in Table 5.12, we observe that there is no *ideal* graph robustness metric to be the objective function for adding new links. However, adding links to balance link-betweenness shows the highest accuracy among all studied robustness functions in providing the best network resilience against centrality-based attacks.

5.9 Summary

In this chapter, we started by presenting our evaluation of three improvement algorithms: algebraic connectivity, path diversity, and balancing centrality. Then, we presented two comparison evaluations: spectral metric comparison and comprehensive improvement.

For algebraic connectivity improvement, we applied our algorithm, described in Section 4.1.1, to physical-level topologies of the several backbone providers. The results showed trade-offs between improving algebraic connectivity and minimizing cost, from which a cost-efficient set of link addition can be chosen based on the value of γ. Moreover, we applied centrality-based attacks on the non- and improved graphs and study their resilience in terms of flow robustness. We showed that graphs with higher algebraic connectivities have mostly higher flow robustness values, which means that they are more resilient

For path diversity improvement, we applied our algorithm to three service provider physical-level topologies. Using the flow robustness graph metric, the path diversity improved graphs were compared to both lowest degree non- and improved graphs as they were attacked using node removal based on highest node centrality graph metrics. The path diversity improved graphs show better resilience to these attacks compared to the lowest degree non- and improved graphs.

For balancing centrality improvement, we evaluated our algorithm that minimizes the node centrality variance of a given graph based on three centrality functions: node betweenness, closeness, and node degree. Then, we applied this algorithm using each function on three physical graphs and study the variance minimization and cost incurred while adding the links. We studied the resilience of the non- and improved graphs using the centrality attacks via the same centrality functions. For each attack, we study the flow robustness of each non- and improved graph. Overall, the results showed the degree-improved graphs outperform the other two improvement methods for these physical graphs. Then, with a similar outcome, the betweenness comes in second, and the weakest improvement method is closeness minimization.

For spectral graph metrics evaluation and improvement, we evaluated a set of graph

spectra robustness metrics: algebraic connectivity, spectral gap, natural connectivity, weighted spectrum, and network criticality. Among these robustness metrics, we showed that network criticality is the best metric to measure baseline, Waxman and Gabriel random graphs resilience against centrality-based attacks. For Gilbert random graphs, we showed that none of the compared metrics can predicate graph robustness with consistency. Moreover, we showed that network criticality yields the best results in improving a given real-world physical graph by adding a specific number of links via a greedy algorithm. Similarly, the algebraic connectivity improved graphs yielded the second best results in terms of resilience against centrality-based attacks.

For our comprehensive comparison evaluation, we evaluated a set of graph robustness metrics to measure their accuracy in predicting network resilience against centrality-based attacks via baseline and random graphs. For baseline graphs, we showed that the path diversity metric has a high accuracy in predicting network resilience. Generally, the variance of node-betweenness centrality has the highest accuracy. For Waxman graphs, which resemble logical-level networks, the variance of node-betweenness centrality metric has the highest accuracy for degree- and closeness-based attacks while network criticality $\hat{\tau}$ and effective graph resistance C^* have better results for betweenness-based attack. All path diversity, network criticality, and effective graph resistance have a high accuracy in measure network resilience centrality-based attacks for Gabriel graphs, which resemble physical-level networks.

For our comprehensive improvement comparison evaluation, we applied our improvement algorithm with each presented graph metric as an objective function. Using three datasets, we showed how each improved graph behaves against centrality-based attacks. Our results indicate that network criticality, link betweenness, and node betweenness-balanced graphs are generally more resilient to centrality-based attacks.

Table 5.12: Resilience values via state model evaluation

Graph	Degree-based	Closeness-based	Betweenness-based
Level 3-λ_2-improved	9.51	7.94	7.40
Level 3-As-improved	8.66	9.48	5.81
Level 3-CC-improved	6.95	6.92	5.19
Level 3-$\Delta\lambda$-improved	7.73	6.21	5.84
Level 3-$\bar{d_{ij}}$-improved	7.42	6.58	5.17
Level 3-$\hat{\tau}$-improved	10.46	10.02	8.28
Level 3-$\bar{\lambda}$-improved	8.92	6.72	5.17
Level 3-$\sigma^2_{C_C}$-improved	10.44	9.77	7.32
Level 3-$\sigma^2_{C_D}$-improved	8.72	7.75	7.12
Level 3-$\sigma^2_{C_{B_l}}$-improved	**11.96**	**11.45**	**9.27**
Level 3-$\sigma^2_{C_{B_n}}$-improved	10.38	11.43	8.52
Level 3-WS-improved	8.93	8.10	6.54
Level 3-TGD-improved	8.08	8.28	6.78
CORONET-λ_2-improved	9.37	8.16	7.39
CORONET-As-improved	9.19	6.58	6.04
CORONET-CC-improved	9.03	6.88	5.84
CORONET-$\Delta\lambda$-improved	9.31	7.11	6.71
CORONET-$\bar{d_{ij}}$-improved	**9.57**	7.04	6.96
CORONET-$\hat{\tau}$-improved	9.55	8.30	6.85
CORONET-$\bar{\lambda}$-improved	9.52	7.09	6.72
CORONET-$\sigma^2_{C_C}$-improved	9.41	7.36	6.53
CORONET-$\sigma^2_{C_D}$-improved	9.36	6.59	6.26
CORONET-$\sigma^2_{C_{B_l}}$-improved	**9.55**	**8.77**	**7.35**
CORONET-$\sigma^2_{C_{B_n}}$-improved	9.52	**8.82**	6.93
CORONET-WS-improved	8.30	6.43	6.31
CORONET-TGD-improved	9.50	9.21	7.07
Internet2-λ_2-improved	6.80	5.83	**6.07**
Internet2-As-improved	6.93	4.64	4.36
Internet2-CC-improved	4.43	4.48	4.08
Internet2-$\Delta\lambda$-improved	6.16	4.54	3.46
Internet2-$\bar{d_{ij}}$-improved	6.47	5.21	3.66
Internet2-$\hat{\tau}$-improved	6.58	5.32	5.26
Internet2-$\bar{\lambda}$-improved	4.68	4.34	3.63
Internet2-$\sigma^2_{C_C}$-improved	5.73	4.71	4.58
Internet2-$\sigma^2_{C_D}$-improved	**7.05**	4.81	4.55
Internet2-$\sigma^2_{C_{B_l}}$-improved	**7.05**	5.78	5.31
Internet2-$\sigma^2_{C_{B_n}}$-improved	**7.05**	**7.05**	5.29
Internet2-WS-improved	4.10	5.04	4.26
Internet2-TGD-improved	6.85	5.39	4.69

Chapter 6

Conclusions and Future Work

This dissertation presents models to un- and weighted graphs for communication networks. Using these models, we study the network resilience against network random failures and targeted attacks. In addition, we investigate several graph robustness metrics presented in the literature and evaluate their accuracy to measure network resilience against such attacks. Furthermore, we design several greedy algorithms to improve a given graph resilience via adding a set of links. Finally, using a dataset of real-world and synthetic graphs, we the resilient of the improved graphs of our dataset and compare their robustness against centrality-based attacks.

6.1 Conclusions

In Chapter 3, we presented our dataset, which includes baseline, random, and real-world graphs. Then, we introduced our graph robustness metrics and how they capture balanced-centrality of a given graph. We presented our weighted flow robustness and showed how it is used to capture network resilience. We also introduced several centrality-based attack models and how they are used to measure network resilience. In addition, we discussed how our model converts unweighted graphs to weighted graphs using node

populations and link betweenness. Finally, we presented how we use flow robustness to measure network resilience against centrality attacks.

In Chapter 4, we presented four improvement algorithms to improve network resilience against targeted attacks. First, the algebraic-connectivity improvement algorithm was presented to add links to improve maximize algebraic connectivity, which improves the resilience against graph partitioning. Next, the path diversity-improvement algorithm was presented to improve network path diversity in terms of increasing the number of disjoint paths. Third, the balancing-centrality-improvement algorithm was introduced to add links to balance a given graph centrality such as degree, closeness, and betweenness. Finally, the comprehensive comparison algorithm was presented to add links to a given graph via a specified robustness function.

In Chapter 5, we presented several evaluation improvement algorithms: algebraic connectivity, path diversity, balancing centrality, spectral-metric comparison, and comprehensive improvement. We showed trade-offs between improving algebraic connectivity and minimizing cost, from which a cost-efficient set of link additions can be chosen based on the value of γ. For path diversity improvement, our results showed that the path diversity improved graphs have better resilience to these attacks compared to the lowest degree non- and improved graphs. For balancing centrality improvement, the results show the degree-improved graphs outperform the other two improvement methods for these physical graphs. Then, with a similar outcome, the betweenness comes in second, and the weakest improvement method is closeness minimization. For spectral graph metrics evaluation and improvement, we evaluate a set of graph spectra robustness metrics: algebraic connectivity, spectral gap, natural connectivity, weighted spectrum, and network criticality. Among these robustness metrics, we show that network criticality is the best metric to measure baseline, Waxman and Gabriel random graphs resilience against centrality-based attacks. For Gilbert random graphs, we show that there is none of the compared

metrics can predicate graph robustness with consistency. Moreover, we demonstrate that network criticality yields the best results in improving a given real-world physical graph by adding a specific number of links via a greedy algorithm. Similarly, the algebraic connectivity improved graphs yield the second-best results in terms of resilience against centrality-based attacks.

For our comprehensive comparison evaluation, we studied a set of graph robustness metrics to measure their accuracy in predicting network resilience against centrality-based attacks via baseline and random graphs. For baseline graphs, we showed that the path diversity metric has a high accuracy in predicting network resilience. Generally, the variance of node-betweenness centrality has the highest accuracy to measure network resilience. For Waxman graphs, which resemble logical-level networks, the variance of node-betweenness centrality metric has the highest accuracy for degree- and closeness-based attacks while network criticality $\hat{\tau}$ and effective graph resistance C^* have better results for the betweenness-based attack. All path diversity, network criticality, and effective graph resistance have high accuracy in measuring network resilience against centrality-based attacks for Gabriel graphs, which resemble physical-level networks. For our studies of comprehensive improvement comparison, we applied our improvement algorithm with each presented graph metric as an objective function. Using three datasets, we showed how each improved graph behaves against centrality-based attacks. Our results indicated that network design based on network criticality, link betweenness and node betweenness balanced graphs are generally more resilient to centrality-based attacks.

For the resilience state model evaluation, we applied on three networks: Level 3, Internet2, and CORONET. The resilience state model is discussed in Section 2.3. By studying the results for all networks, we observe that there is no *ideal* graph robustness metric to be the objective function for adding new links. However, adding links to balance link-betweenness shows the highest consistency among all studied robustness functions

in providing the best network resilience against centrality-based attacks.

6.2 Future Work

In this research, we focused on physical level networks. For our future work, we plan to study the improvement of logical level networks and other overlay networks. The cost of adding physical layer links is dominated by the link length. However, the cost in logical level networks is related to the number of ports. Multilevel evaluation and improvement can be more efficient for overall network performance. However, combining logical and physical network levels in our improvement algorithms can lead to more sophisticated research problems. In this research, we consider node locations for new link cost. We plan to investigate new methods to improve graph diversity by balancing the node locations to avoid correlated geographic failures. Moreover, adding new nodes is a similar problem, which requires further investigation.

Here, we focused on evaluating a given network by applying centrality-based attacks. In the future, we plan to investigate other challenges such as correlated geographic failures and random failures. Our measure for network resilience is the sum of flow robustness as nodes are attacked, which is a graph theoretic model. We plan to evaluate and compare improved graphs using the ns-3 [111] network simulator to study other performance parameters such as packet delivery and end-to-end delay in a more realistic environment.

Bibliography

[1] James P.G. Sterbenz, Egemen K. Çetinkaya, Mahmood A. Hameed, Abdul Jabbar, Qian Shi, and Justin P. Rohrer. Evaluation of Network Resilience, Survivability, and Disruption Tolerance: Analysis, Topology Generation, Simulation, and Experimentation (invited paper). *Telecommunication Systems*, 52(2):705–736, 2013.

[2] Richard L Street, William R Gold, and Timothy R Manning. *Health promotion and interactive technology: Theoretical applications and future directions*. Routledge, 2013.

[3] Dongsong Zhang, J Leon Zhao, Lina Zhou, and Jay F Nunamaker Jr. Can e-learning replace classroom learning? *Communications of the ACM*, 47(5):75–79, 2004.

[4] Global B2C Ecommerce Sales to Hit $1.5 Trillion This Year Driven by Growth in Emerging Markets - eMarketer.

[5] B. Donnet and T. Friedman. Internet topology discovery: a survey. *IEEE Communications Surveys & Tutorials*, 9(4):56–69, 2007.

[6] James P. G. Sterbenz and David Hutchison. Resilinets: Multilevel resilient and survivable networking initiative wiki. `http://wiki.ittc.ku.edu/resilinets`, April 2006.

[7] James P. G. Sterbenz, David Hutchison, Egemen K. Çetinkaya, Abdul Jabbar, Justin P. Rohrer, Marcus Schöller, and Paul Smith. Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines. *Computer Networks*, 54(8):1245–1265, 2010.

[8] ENISA Virtual Working Group on Network Providers Resilience Measures. Network resilience and security: Challenges and measures. Technical Report WP 2009 – WPK 1.2 VWG 1, ENISA – European Network and Information Security Agency, December 2009.

[9] Egemen K. Çetinkaya, Mohammed J. F. Alenazi, Andrew M. Peck, Justin P. Rohrer, and James P. G. Sterbenz. Multilevel Resilience Analysis of Transportation and Communication Networks. *Springer Telecommunication Systems Journal*, 2013. (accepted in July 2013).

[10] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[11] Justin P. Rohrer, Abdul Jabbar, and James P. G. Sterbenz. Path Diversification: A Multipath Resilience Mechanism. In *Proceedings of the IEEE 7th International Workshop on the Design of Reliable Communication Networks (DRCN)*, pages 343–351, Washington, DC, October 2009.

[12] Kapali P Eswaran and R Endre Tarjan. Augmentation problems. *SIAM Journal on Computing*, 5(4):653–665, 1976.

[13] Damon Mosk-Aoyama. Maximum algebraic connectivity augmentation is NP-hard. *Operations Research Letters*, 36(6):677–679, 2008.

[14] Egemen K. Çetinkaya, Andrew M. Peck, and James P. G. Sterbenz. Flow Robustness of Multilevel Networks. In *Proceedings of the 9th IEEE/IFIP International Conference on the Design of Reliable Communication Networks (DRCN)*, pages 274–281, Budapest, March 2013.

[15] Robert S. Wilkov. Analysis and design of reliable computer networks. *IEEE Transactions on Communications*, 20(3):660–678, 1972.

[16] Howard Frank and Wushow Chou. Topological optimization of computer networks. *Proceedings of the IEEE*, 60(11):1385–1397, 1972.

[17] Michael O. Ball. Complexity of network reliability computations. *Networks*, 10(2):153–165, 1980.

[18] Michael O. Ball. Computational complexity of network reliability analysis: An overview. *IEEE Transactions on Reliability*, 35(3):230–239, 1986.

[19] Samir Khuller and Balaji Raghavachari. Graph and Network Algorithms. *ACM Comput. Surv.*, 28(1):43–45, 1996.

[20] H. Noltemeier, H.-C. Wirth, and S. O. Krumke. Network Design and Improvement. *ACM Comput. Surv.*, 31(3es):1–5, 1999.

[21] Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1982.

[22] Robert Endre Tarjan. *Data Structures and Network Algorithms*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1983.

[23] William J. Cook, William H. Cunningham, William R. Pulleyblank, and Alexander Schrijver. *Combinatorial Optimization*. John Wiley & Sons, Inc., New York, NY, USA, 1998.

[24] Bernhard Korte and Jens Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer Publishing Company, Incorporated, 4th edition, 2007.

[25] Alina Beygelzimer, Geoffrey Grinstein, Ralph Linsker, and Irina Rish. Improving network robustness by edge modification. *Physica A: Statistical Mechanics and its Applications*, 357(3–4):593–612, 2005.

[26] Christian M. Schneider, André A. Moreira, José S. Andrade, Shlomo Havlin, and Hans J. Herrmann. Mitigation of malicious attacks on networks. *Proceedings of the National Academy of Sciences*, 108(10):3838–3841, 2011.

[27] Bogdan Danila, Yong Yu, John A. Marsh, and Kevin E. Bassler. Optimal transport on complex networks. *Phys. Rev. E*, 74:046106, Oct 2006.

[28] Bogdan Danila, Yong Yu, John A. Marsh, and Kevin E. Bassler. Transport optimization on complex networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 17(2), 2007.

[29] Michiel Hazewinkel. Greedy algorithm. `http://www.encyclopediaofmath.org/index.php?title=Greedy_algorithm`.

[30] Thomas H. Cormen, Clifford Stein, Ronald L. Rivest, and Charles E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition, 2001.

[31] George B. Dantzig. Discrete-variable extremum problems. *Operations Research*, 5(2):266–288, 1957.

[32] Justin P. Rohrer, Abdul Jabbar, and James P.G. Sterbenz. Path Diversification for Future Internet End-to-End Resilience and Survivability. *Springer Telecommunication Systems*, 56(1):49–67, May 2014.

[33] Huijuan Wang and Piet Van Mieghem. Algebraic connectivity optimization via link addition. In *Proceedings of the 3rd ICST International Conference on Bio-Inspired Models of Network, Information and Computing Sytems (BIONETICS)*, pages 22:1–22:8, Hyogo, Japan, November 2008.

[34] William Liu, Harsha Sirisena, Krzysztof Pawlikowski, and Allan McInnes. Utility of algebraic connectivity metric in topology design of survivable networks. In *Proceedings of the 7th IEEE International Workshop on Design of Reliable Communication Networks (DRCN)*, pages 131–138, Washington, DC, October 2009.

[35] Ali Sydney, Caterina Scoglio, and Don Gruenbacher. Optimizing algebraic connectivity by edge rewiring. *Applied Mathematics and Computation*, 219(10):5465–5479, 2013.

[36] A. Jamaković and S. Uhlig. On the relationship between the algebraic connectivity and graph's robustness to node and link failures. In *Proceedings of the 3rd EuroNGI Conference on Next Generation Internet Networks*, pages 96–102, Trondheim, May 2007.

[37] D. Fay, H. Haddadi, A. Thomason, A.W. Moore, R. Mortier, A. Jamakovic, S. Uhlig, and M. Rio. Weighted Spectral Distribution for Internet Topology Analysis: Theory and Applications. *IEEE/ACM Transactions on Networking*, 18(1):164–176, 2010.

[38] Xuelian Long, David Tipper, and Teresa Gomes. Measuring the survivability of networks to geographic correlated failures. *Optical Switching and Networking*, 14, Part 2(0):117–133, 2014.

[39] Jun Wu, Mauricio Barahona, Yue-Jin Tan, and Hong-Zhong Deng. Spectral measure of structural robustness in complex networks. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 41(6):1244–1252, 2011.

[40] Ali Tizghadam and Alberto Leon-Garcia. Autonomic traffic engineering for network robustness. *Selected Areas in Communications, IEEE Journal on*, 28(1):39–50, 2010.

[41] Alireza Bigdeli, Ali Tizghadam, and Alberto Leon-Garcia. Comparison of network criticality, algebraic connectivity, and other graph metrics. In *Proceedings of the 1st Annual Workshop on Simplifying Complex Network for Practitioners*, page 4. ACM, 2009.

[42] E Estrada. Network robustness to targeted attacks. the interplay of expansibility and degree distribution. *The European Physical Journal B-Condensed Matter and Complex Systems*, 52(4):563–574, 2006.

[43] A Yazdani, R Appiah Otoo, and P Jeffrey. Resilience enhancing expansion strategies for water distribution systems: A network theory approach. *Environmental Modelling & Software*, 26(12):1574–1582, 2011.

[44] Xiangrong Wang, Evangelos Pournaras, Robert E Kooij, and Piet Van Mieghem. Improving robustness of complex networks via the effective graph resistance. *The European Physical Journal B*, 87(9):1–12, 2014.

[45] John Hopcroft and Robert Tarjan. Algorithm 447: efficient algorithms for graph manipulation. *Communications of the ACM*, 16(6):372–378, 1973.

[46] Stephen P. Borgatti. Centrality and Network Flow. *Social Networks*, 27(1):55–71, 2005.

[47] Stephen P. Borgatti and Martin G. Everett. A Graph-Theoretic Perspective on Centrality. *Social Networks*, 28(4):466–484, 2006.

[48] L. da F. Costa, F. A. Rodrigues, G. Travieso, and P. R. Villas Boas. Characterization of complex networks: A survey of measurements. *Advances in Physics*, 56(1):167–242, 2007.

[49] T. Opsahl, F. Agneessens, and J. Skvoretz. Node centrality in weighted networks: Generalizing degree and shortest paths. *Social Networks*, 32(3):245–251, 2010.

[50] M. E. J. Newman. Scientific collaboration networks. ii. shortest paths, weighted networks, and centrality. *Phys. Rev. E*, 64(1):16132, 2001.

[51] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959. 10.1007/BF01386390.

[52] Robert W. Floyd. Algorithm 97: Shortest path. *ACM Communications*, 5(6):345, 1962.

[53] Linton C. Freeman. Centrality in Social Networks Conceptual Clarification. *Social Networks*, 1(3):215–239, 1978–1979.

[54] Alain Barrat, Marc Barthelemy, Romualdo Pastor-Satorras, and Alessandro Vespignani. The architecture of complex weighted networks. *Proceedings of the National Academy of Sciences of the United States of America*, 101(11):3747–3752, 2004.

[55] M. E. J. Newman. Assortative mixing in networks. *Phys. Rev. Lett.*, 89(20):208701, October 2002.

[56] Linton C. Freeman. A Set of Measures of Centrality Based on Betweenness. *Sociometry*, 40(1):35–41, 1977.

[57] Ulrik Brandes. A faster algorithm for betweenness centrality*. *Journal of Mathematical Sociology*, 25(2):163–177, 2001.

[58] M. E. J. Newman. *Networks: An Introduction*, chapter 7, page 199. Oxford University Press, 1st edition, 2010.

[59] U Kang, Spiros Papadimitriou, Jimeng Sun, and Hanghang Tong. Centralities in large networks: Algorithms and observations. In *SDM*, volume 2011, pages 119–130. SIAM, 2011.

[60] Fan R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

[61] P. Van Mieghem. *Graph Spectra for Complex Networks*. Cambridge University Press, 2011.

[62] Norman Biggs. *Algebraic Graph Theory*. Cambridge University Press, 2nd edition, 1993.

[63] Andries E. Brouwer and Willem H. Haemers. *Spectra of Graphs*. Springer New York, 2012.

[64] Dragoš Cvetković, Peter Rowlinson, and Slobodan Simić. *An Introduction to the Theory of Graph Spectra*. London Mathematical Society, 2009.

[65] Bojan Mohar and Y Alavi. The laplacian spectrum of graphs. *Graph theory, combinatorics, and applications*, 2:871–898, 1991.

[66] Egemen K. Çetinkaya, Mohammed J. F. Alenazi, Justin P. Rohrer, and James P. G. Sterbenz. Topology Connectivity Analysis of Internet Infrastructure Using Graph

Spectra. In *Proceedings of the 4th IEEE/IFIP International Workshop on Reliable Networks Design and Modeling (RNDM)*, pages 752–758, St. Petersburg, October 2012.

[67] Mohammed J. F. Alenazi, Egemen K. Çetinkaya, and James P. G. Sterbenz. Network Design and Optimisation Based on Cost and Algebraic Connectivity. In *Proceedings of the 5th IEEE/IFIP International Workshop on Reliable Networks Design and Modeling (RNDM)*, pages 193–200, Almaty, September 2013.

[68] Ali Tizghadam and Alberto Leon-Garcia. Betweenness centrality and resistance distance in communication networks. *Network, IEEE*, 24(6):10–16, 2010.

[69] John G. Apostolopoulos and Mitchell D. Trott. Path Diversity for Enhanced Media Streaming. *IEEE Communications Magazine*, 42(8):80–87, 2004.

[70] Aun Haider and Richard Harris. Recovery Techniques in Next Generation Networks. *IEEE Communications Surveys & Tutorials*, 9(3):2–17, 2007.

[71] Junghee Han, David Watson, and Farnam Jahanian. An Experimental Study of Internet Path Diversity. *IEEE Transactions on Dependable and Secure Computing*, 3(4):273–288, 2006.

[72] Jiayue He and Jennifer Rexford. Toward Internet-Wide Multipath Routing. *IEEE Network Magazine*, 22(2):16–21, 2008.

[73] J. W. Suurballe. Disjoint Paths in a Network. *Networks*, 4(2):125–145, 1974.

[74] J. W. Suurballe and R. E. Tarjan. A Quick Method for Finding Shortest Pairs of Disjoint Paths. *Networks*, 14(2):325–336, 1984.

[75] Ramesh Bhandari. Optimal Diverse Routing in Telecommunication Fiber Networks. In *Proceedings of the IEEE INFOCOM*, volume 3, pages 1498–1508, Toronto, June 1994.

[76] Murtaza Motiwala, Megan Elmore, Nick Feamster, and Santosh Vempala. Path Splicing. In *Proceedings of the ACM SIGCOMM*, pages 27–38, Seattle, WA, August 2008.

[77] Xiaowei Yang and David Wetherall. Source Selectable Path Diversity via Routing Deflections. In *Proceedings of the ACM SIGCOMM*, pages 159–170, Pisa, September 2006.

[78] Ariel Orda and Alexander Sprintson. Efficient Algorithms for Computing Disjoint QoS Paths. In *Proceedings of the IEEE INFOCOM*, volume 1, pages 727–738, Hong Kong, March 2004.

[79] Yuchun Guo, Fernando Kuipers, and Piet Van Mieghem. Link-Disjoint Paths for Reliable QoS Routing. *International Journal of Communication Systems*, 16(9):779–798, 2003.

[80] Feng Wang and Lixin Gao. Path Diversity Aware Interdomain Routing. In *Proceedings of the IEEE INFOCOM*, pages 307–315, Rio de Janeiro, April 2009.

[81] Hyang-Won Lee, E. Modiano, and Kayi Lee. Diverse Routing in Networks With Probabilistic Failures. *IEEE/ACM Transactions on Networking*, 18(6):1895–1907, 2010.

[82] Yufei Cheng, Junyan Li, and James P. G. Sterbenz. Path Geo-diversification: Design and Analysis. In *Proceedings of the 5th IEEE/IFIP International Workshop on Reliable Networks Design and Modeling (RNDM)*, Almaty, September 2013.

[83] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley. Multi-Path TCP: A Joint Congestion Control and Routing Scheme to Exploit Path Diversity in the Internet. *IEEE/ACM Transactions on Networking*, 14(6):1260–1271, 2006.

[84] Dahai Xu, Yang Chen, Yizhi Xiong, Chunming Qiao, and Xin He. On the complexity of and algorithms for finding the shortest path with a disjoint counterpart. *IEEE/ACM Transactions on Networking*, 14(1):147–158, 2006.

[85] Chung-Lun Li, S.Thomas McCormick, and David Simchi-Levi. The complexity of finding two disjoint paths with min-max objective function. *Discrete Applied Mathematics*, 26(1):105–115, 1990.

[86] Abdul Jabbar, Hemanth Narra, and James P. G. Sterbenz. An approach to quantifying resilience in mobile ad hoc networks. In *Proceedings of the 8th IEEE International Workshop on the Design of Reliable Communication Networks (DRCN)*, pages 140–147, Krakow, Poland, October 2011.

[87] Abdul Jabbar. *A Framework to Quantify Network Resilience and Survivability*. PhD thesis, The University of Kansas, Lawrence, KS, May 2010.

[88] Edgar N Gilbert. Random graphs. *The Annals of Mathematical Statistics*, pages 1141–1144, 1959.

[89] Bernard M. Waxman. Routing of Multipoint Connections. *IEEE Journal on Selected Areas in Communications*, 6(9):1617–1622, 1988.

[90] Egemen K. Cetinkaya, Mohammed J.F. Alenazi, Yufei Cheng, Andrew M. Peck, and James P.G. Sterbenz. A comparative analysis of geometric graph models for modelling backbone networks. *Optical Switching and Networking*, 14, Part 2:95 – 106, 2014.

[91] K. Ruben Gabriel and Robert R. Sokal. A New Statistical Approach to Geographic Variation Analysis. *Systematic Zoology*, 18(3):259–278, 1969.

[92] David W. Matula and Robert R. Sokal. Properties of Gabriel Graphs Relevant to Geographic Variation Research and the Clustering of Points in the Plane. *Geographical Analysis*, 12(3):205–222, 1980.

[93] AT&T. `http://www.att.com`.

[94] Sprint. `http://www.sprint.com`.

[95] Internet2. `http://www.internet2.edu`.

[96] KMI Corporation. North American Fiberoptic Long-haul Routes Planned and in Place, 1999.

[97] George Clapp, Ronald A. Skoog, Ann C. Von Lehmen, and Brian Wilson. Management of Switched Systems at 100 Tbps: the DARPA CORONET Program. In *International Conference on Photonics in Switching (PS)*, pages 1–4, Pisa, September 2009.

[98] The Next Generation Core Optical Networks (CORONET). `http://www.darpa.mil/Our_Work/STO/Programs/Dynamic_Multi-Terabit_Core_Optical_Networks_(CORONET).aspx`.

[99] Simon Knight, Hung X. Nguyen, Nickolas Falkner, Rhys Bowden, and Matthew Roughan. The Internet Topology Zoo. *IEEE Journal on Selected Areas in Communications*, 29(9):1765–1775, 2011.

[100] Research and education advanced network new zealand, Feb 2015.

[101] Wikipedia. Mci communications — wikipedia, the free encyclopedia, 2014. [Online; accessed 16-February-2015].

[102] Croatian academic and research network, February 2015.

[103] GÉANT2. `http://www.geant2.net/`, December 2009.

[104] Egemen K. Çetinkaya, Mohammed J. F. Alenazi, Yufei Cheng, Andrew M. Peck, and James P. G. Sterbenz. On the Fitness of Geographic Graph Generators for Modelling Physical Level Topologies. In *Proceedings of the 5th IEEE/IFIP International Workshop on Reliable Networks Design and Modeling (RNDM)*, pages 38–45, Almaty, September 2013.

[105] Karl Pearson. Mathematical contributions to the theory of evolution.–on a form of spurious correlation which may arise when indices are used in the measurement of organs. *Proceedings of the royal society of London*, 60(359-367):489–498, 1896.

[106] C. Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):pp. 72–101, 1904.

[107] Mohammed J.F. Alenazi, Egemen K. Çetinkaya, and James P. G. Sterbenz. Cost-Constrained and Centrality-Balanced network design improvement. In *RNDM'14 - 6th International Workshop on Reliable Networks Design and Modeling (RNDM 2014)*, pages 194 – 201, Barcelona, Spain, Nov 2014.

[108] Mohammed J.F. Alenazi, Egemen K. Çetinkaya, and James P. G. Sterbenz. Cost-Efficient network improvement to achieve maximum path diversity. In *RNDM'14 - 6th International Workshop on Reliable Networks Design and Modeling (RNDM 2014)*, pages 202 – 208, Barcelona, Spain, November 2014.

[109] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring Network Structure, Dynamics, and Function using NetworkX. In *7th Python in Science Conference (SciPy)*, pages 11–15, Pasadena, CA, August 2008.

[110] Level 3 Network Map. http://maps.level3.com.

[111] The ns-3 Network Simulator. http://www.nsnam.org, July 2009.

Appendix A

Dataset Maps

This appendix contains a full set maps for our dataset presented in Section 3.1.

A.1 Unweighted Maps

Figure A.1: AT&T map

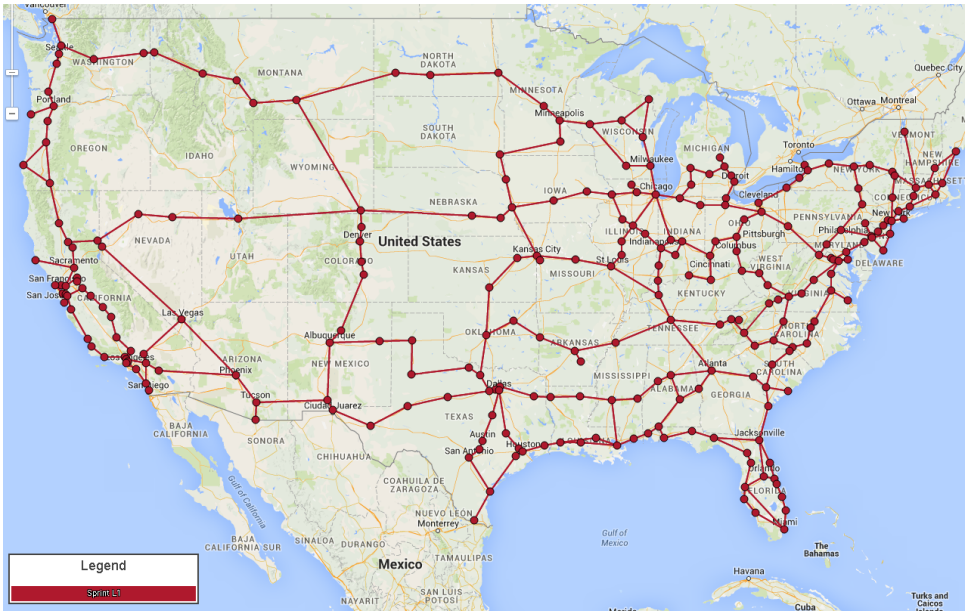

Figure A.2: Sprint map

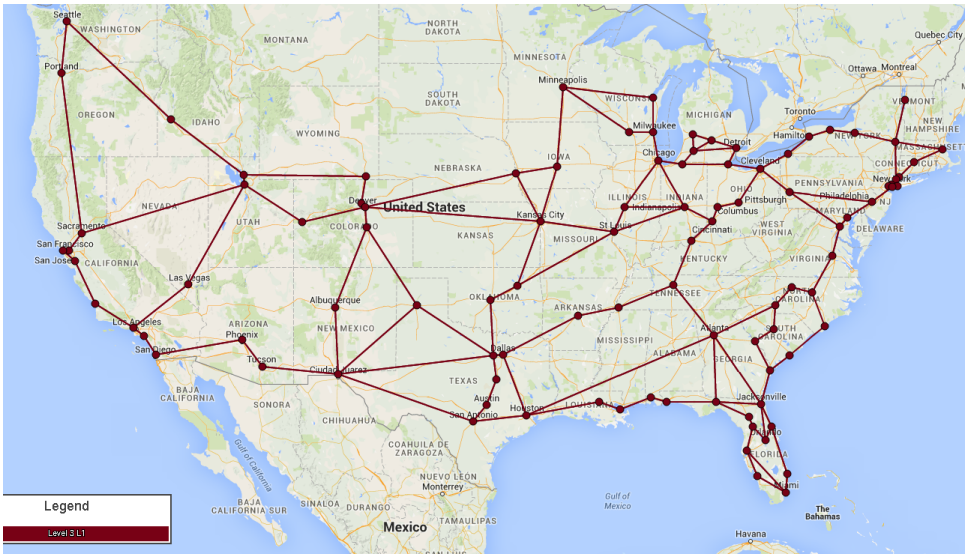

Figure A.3: Level 3 map

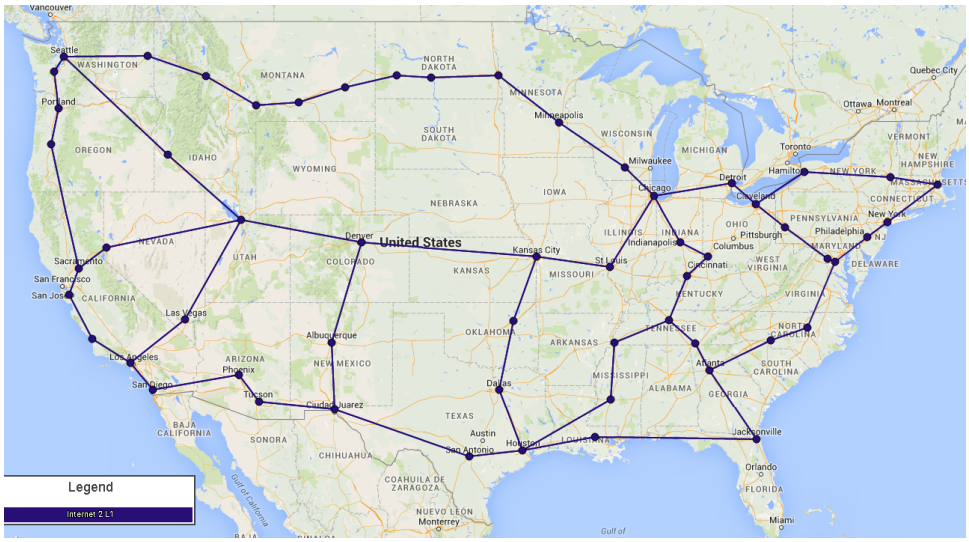

Figure A.4: Internet2 map

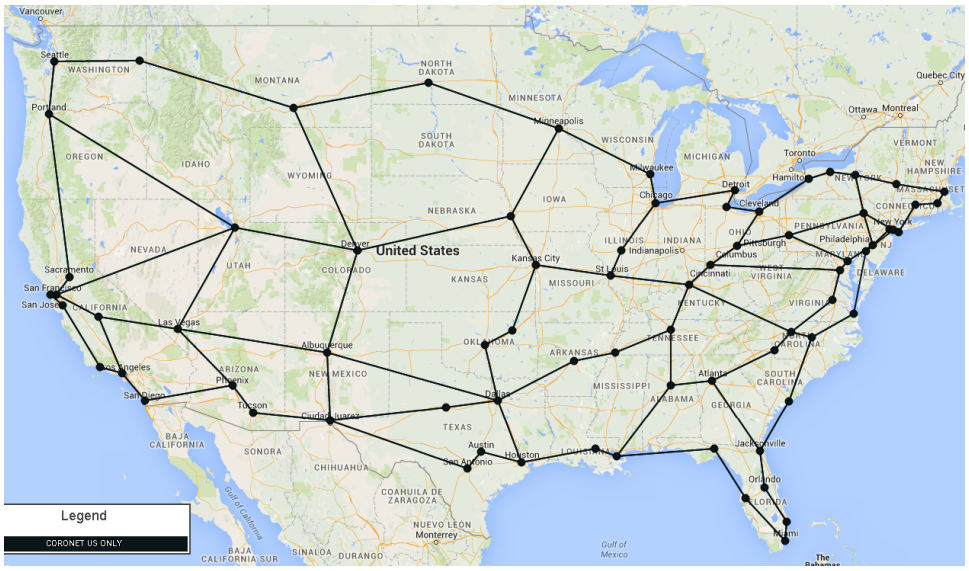

Figure A.5: CORONET map

A.2 Weighted Maps

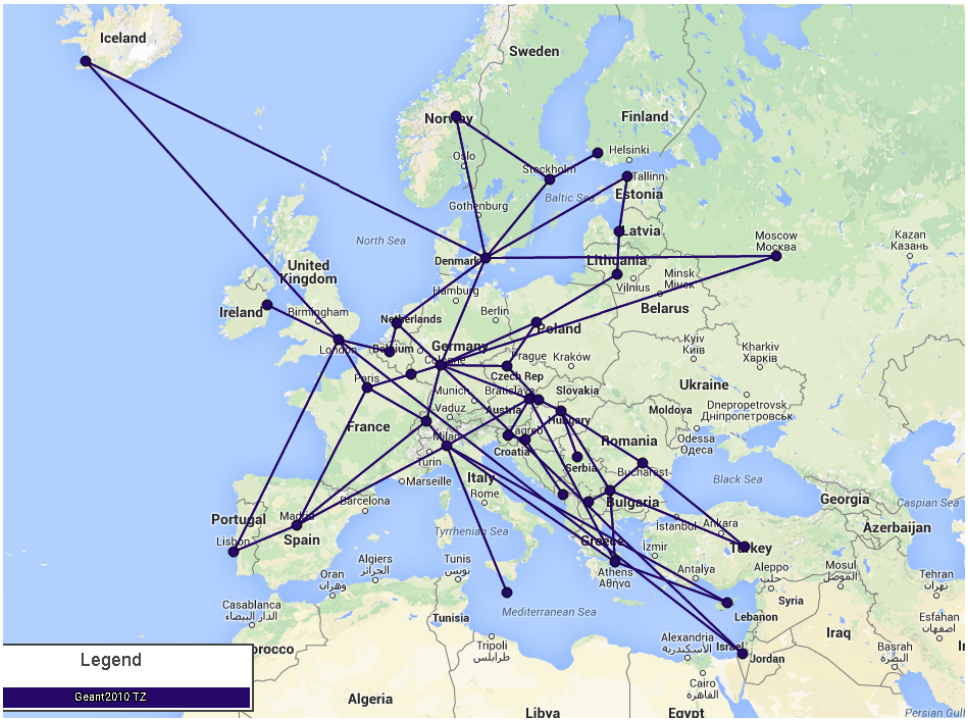

Figure A.6: GÉANT map

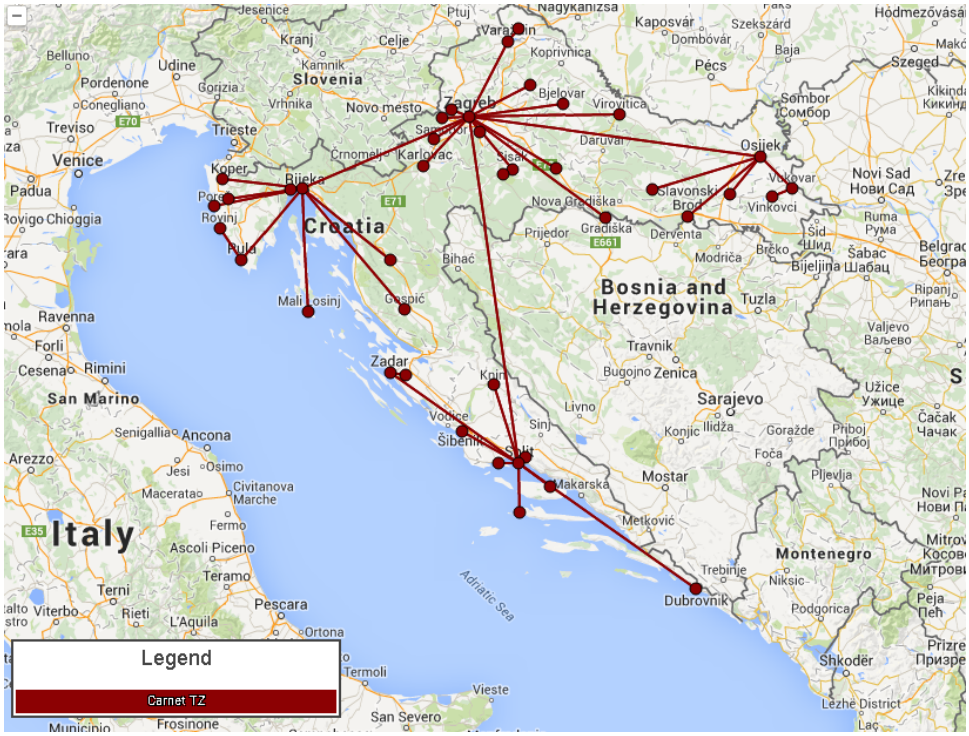

Figure A.7: CARNet map

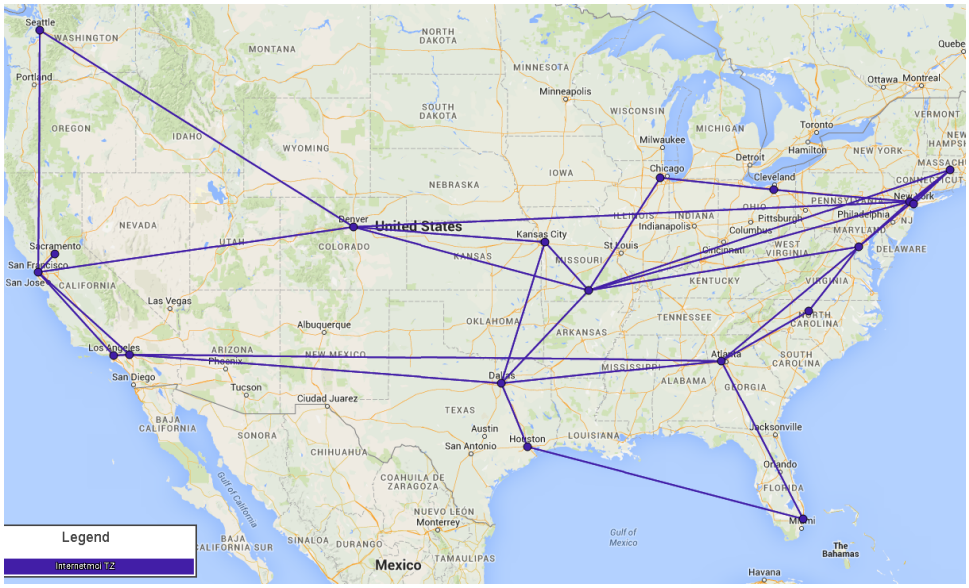

Figure A.8: InternetMCI map

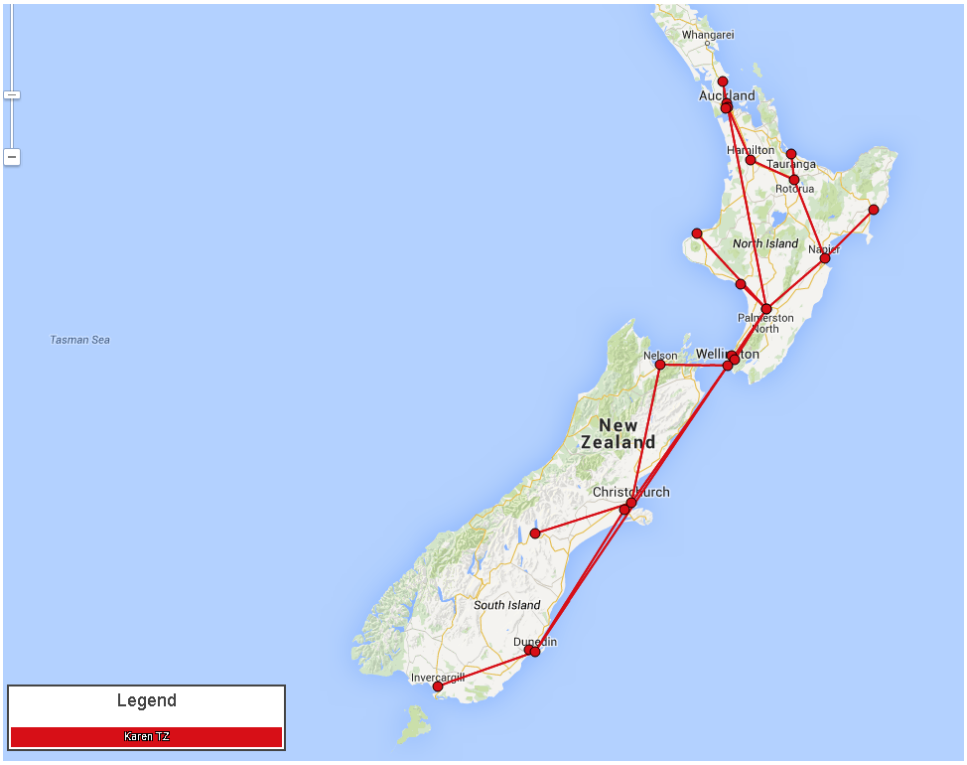

Figure A.9: KAREN map

Appendix B

Graph Improvement Plots

This appendix contains a full set of plots for the topologies used in the analysis of the *graph improvement algorithms*.

B.1 Graph Improvement via Algebraic Connectivity

B.1.1 Algebraic Connectivity Improvement

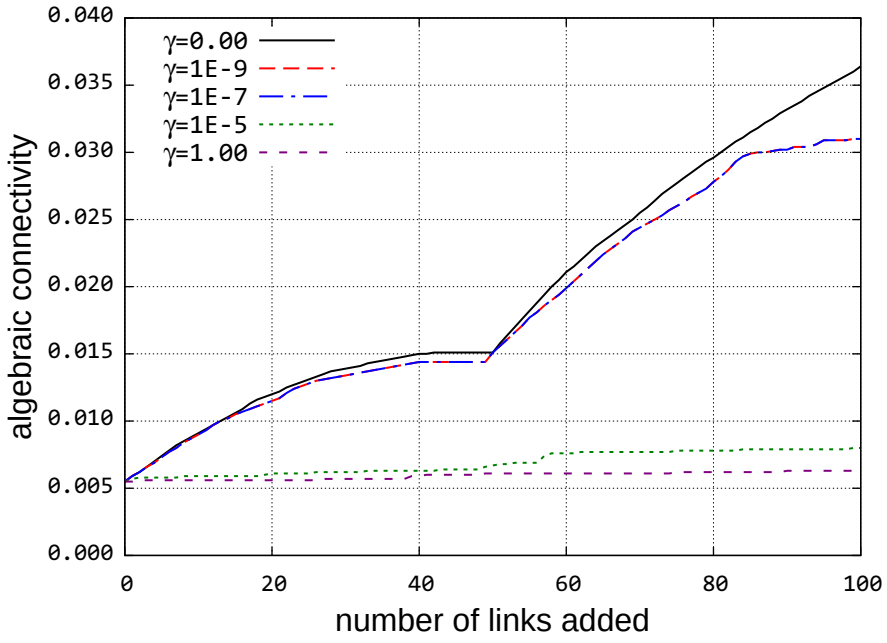

Figure B.1: AT&T connectivity improvement

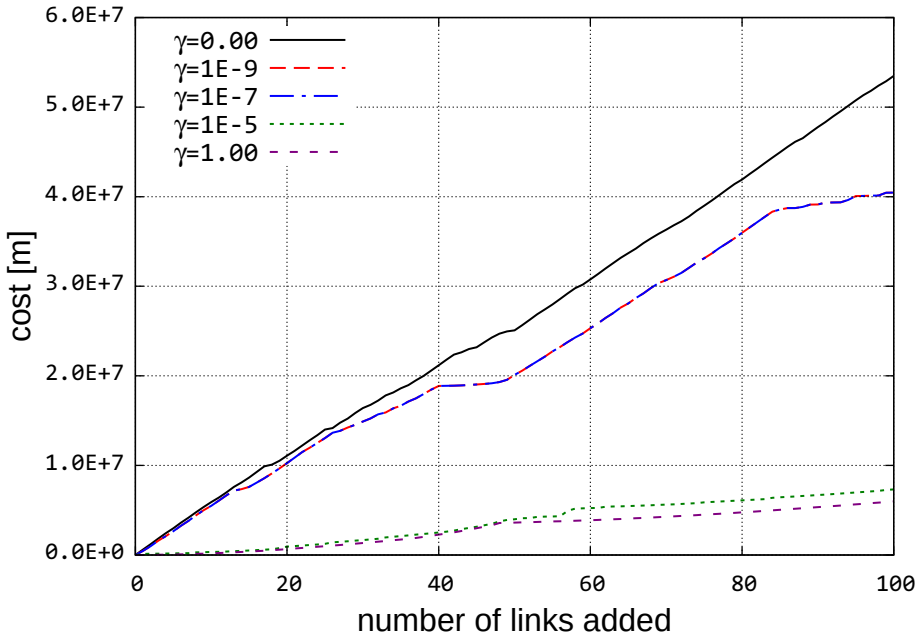

Figure B.2: AT&T cost incurred with adding links

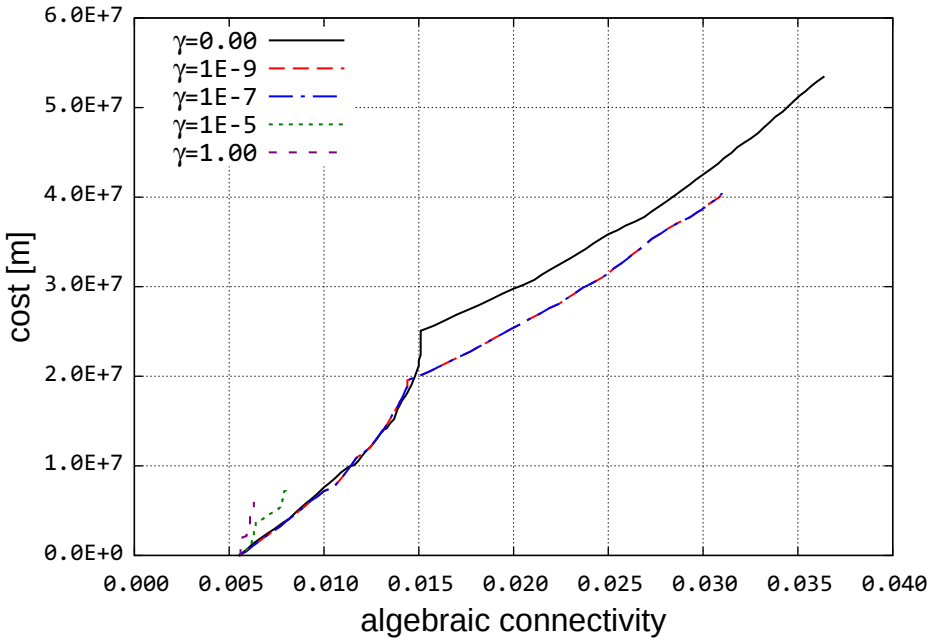

Figure B.3: Connectivity and cost trade-offs for AT&T

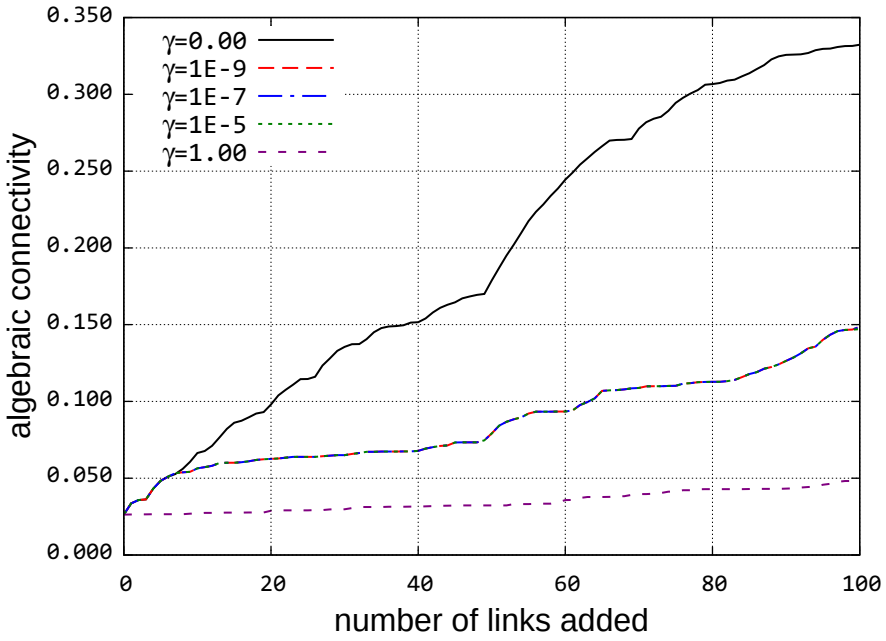

Figure B.4: Level 3 connectivity improvement

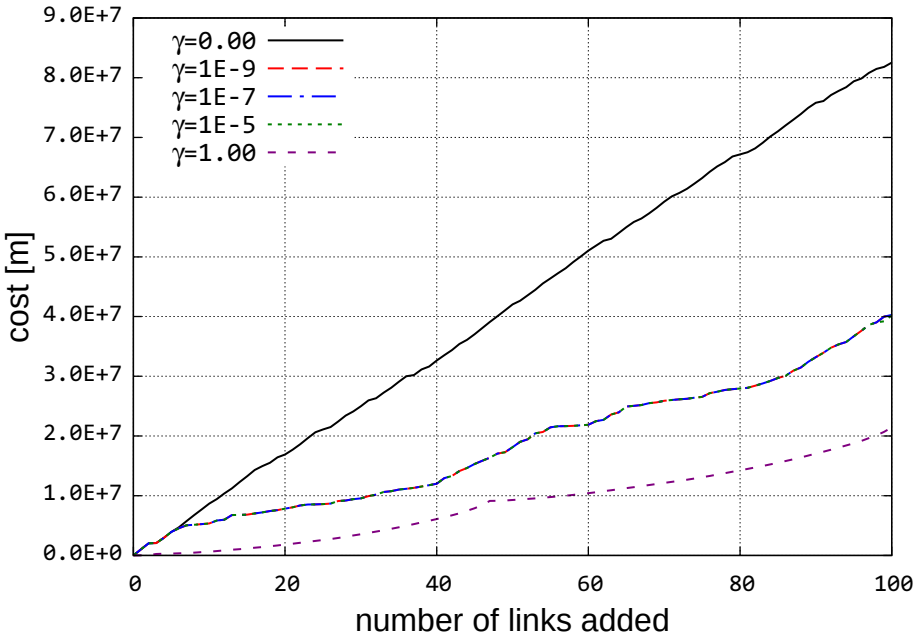

Figure B.5: Level 3 cost incurred with adding links

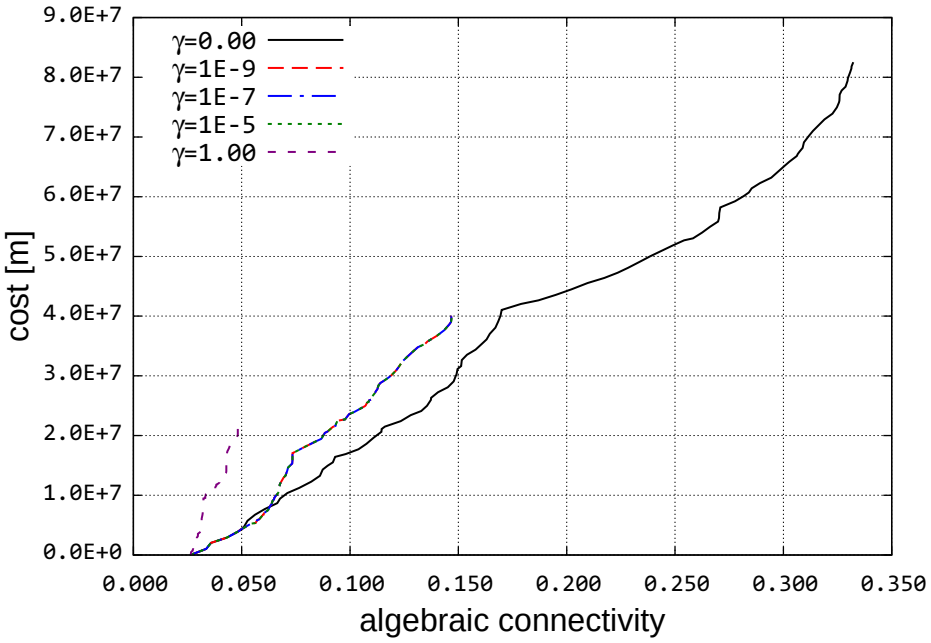

Figure B.6: Connectivity and cost trade-offs for Level 3

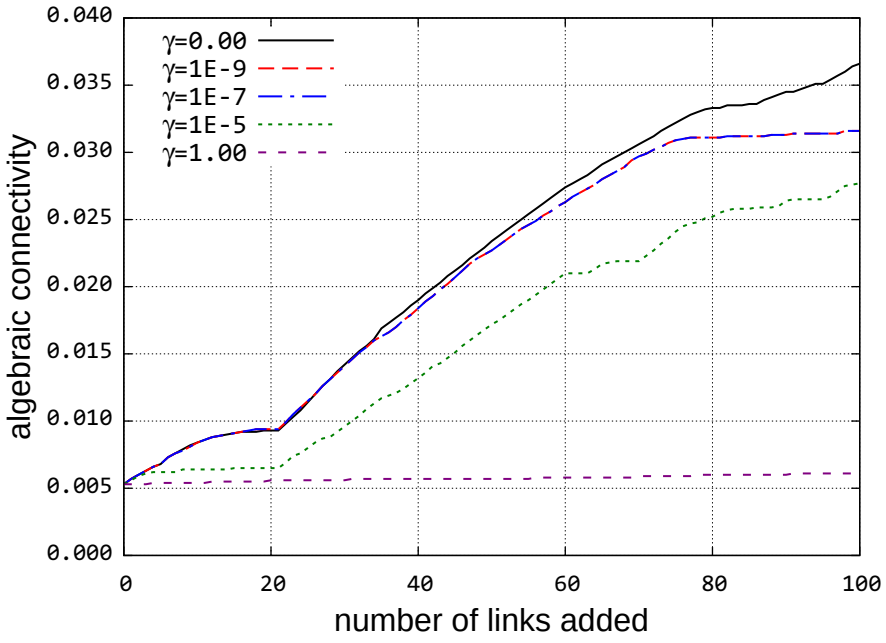

Figure B.7: Sprint connectivity improvement

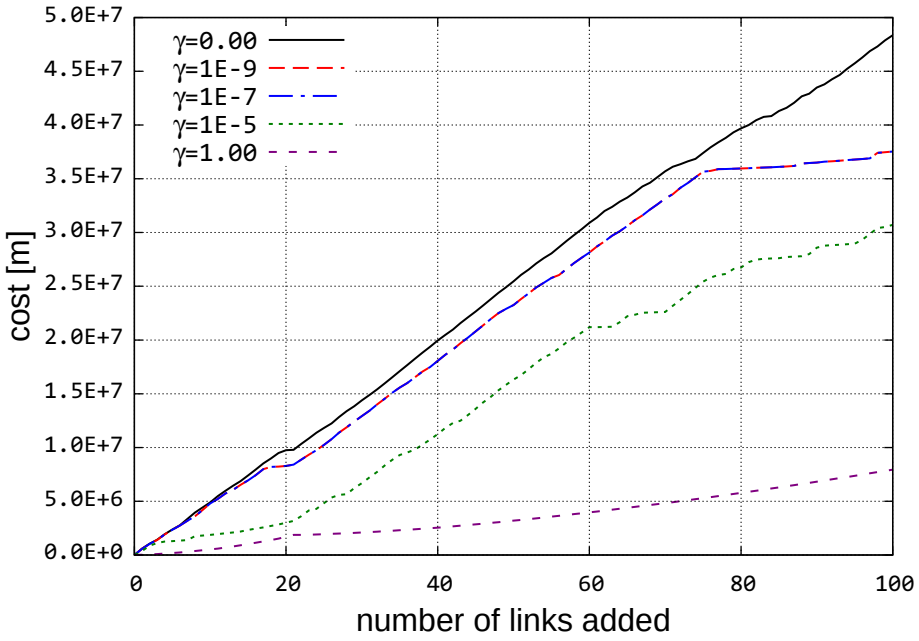

Figure B.8: Sprint cost incurred with adding links

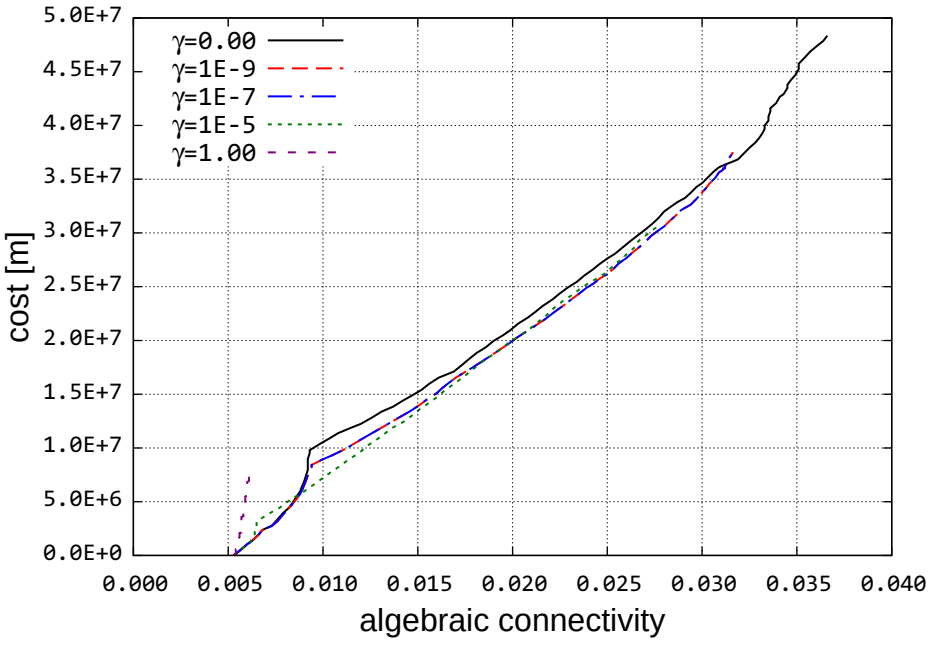

Figure B.9: Connectivity and cost trade-offs for Sprint

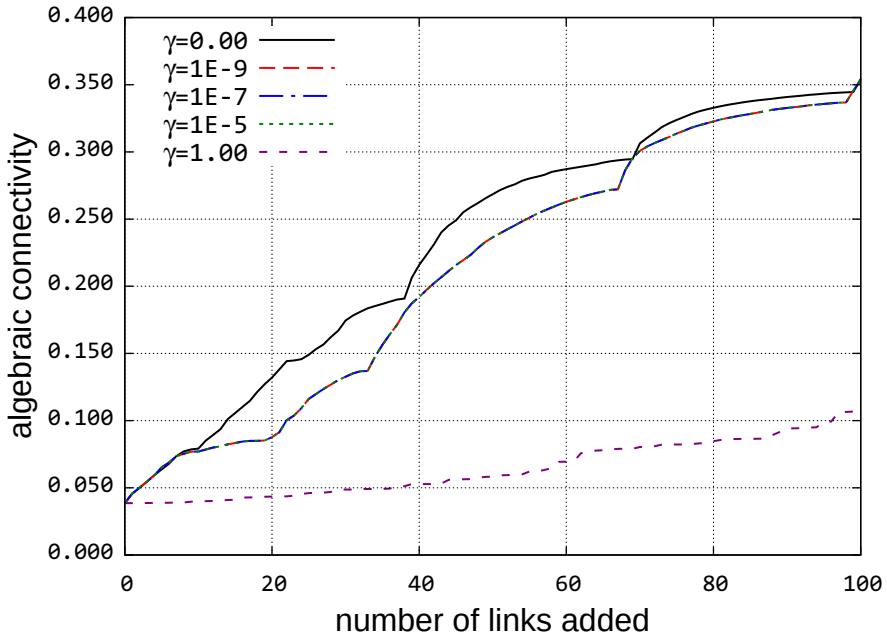

Figure B.10: Internet2 connectivity improvement

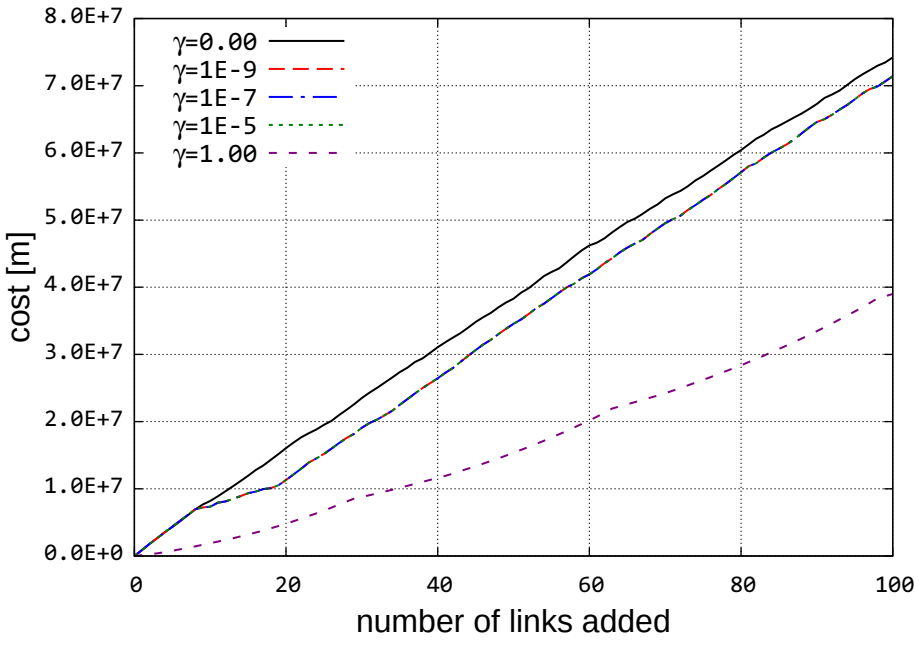

Figure B.11: Internet2 cost incurred with adding links

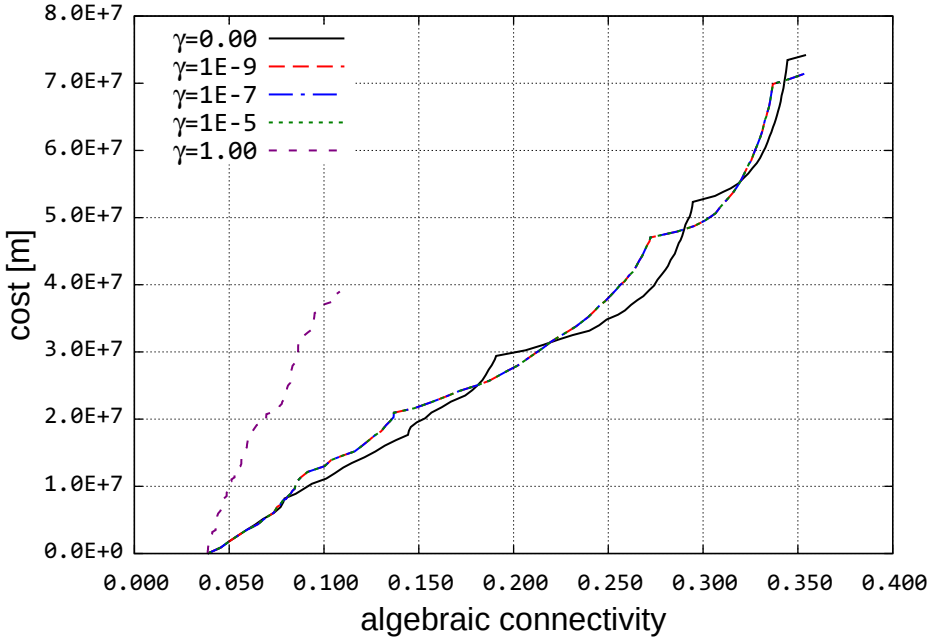

Figure B.12: Connectivity and cost trade-offs for Internet2

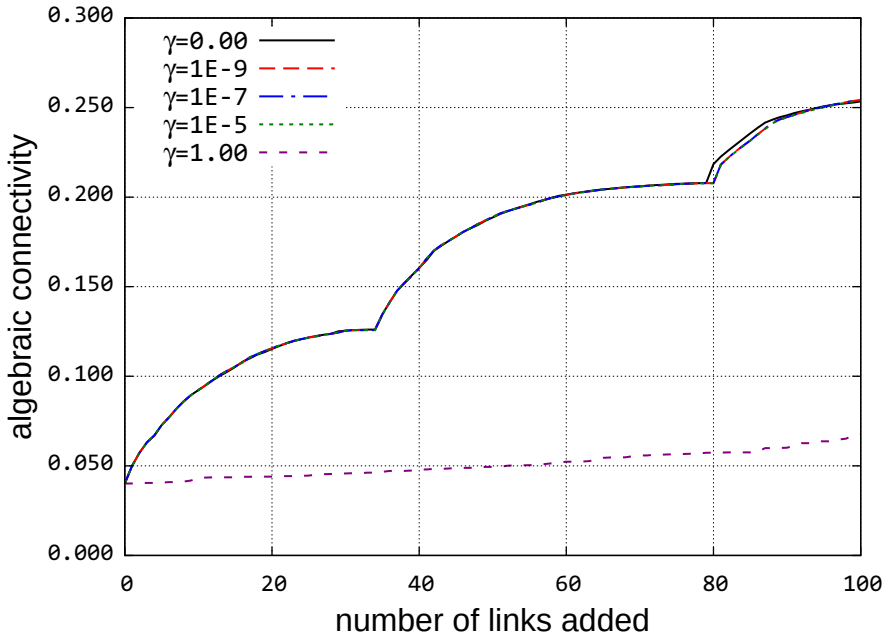

Figure B.13: CORONET connectivity improvement

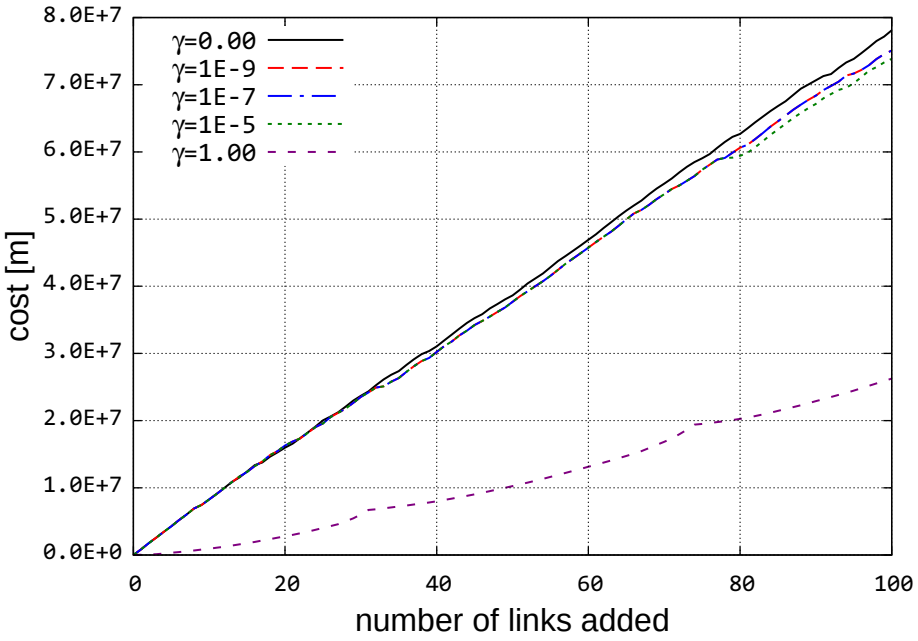

Figure B.14: CORONET cost incurred with adding links

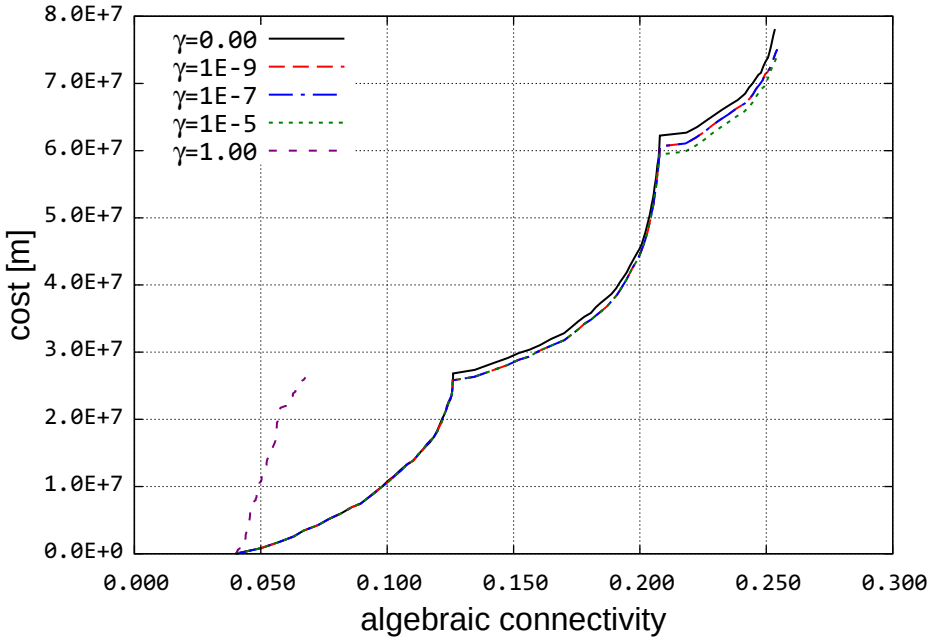

Figure B.15: Connectivity and cost trade-offs for CORONET

B.1.2 Flow Robustness Evaluation of Algebraic Connectivity Graphs

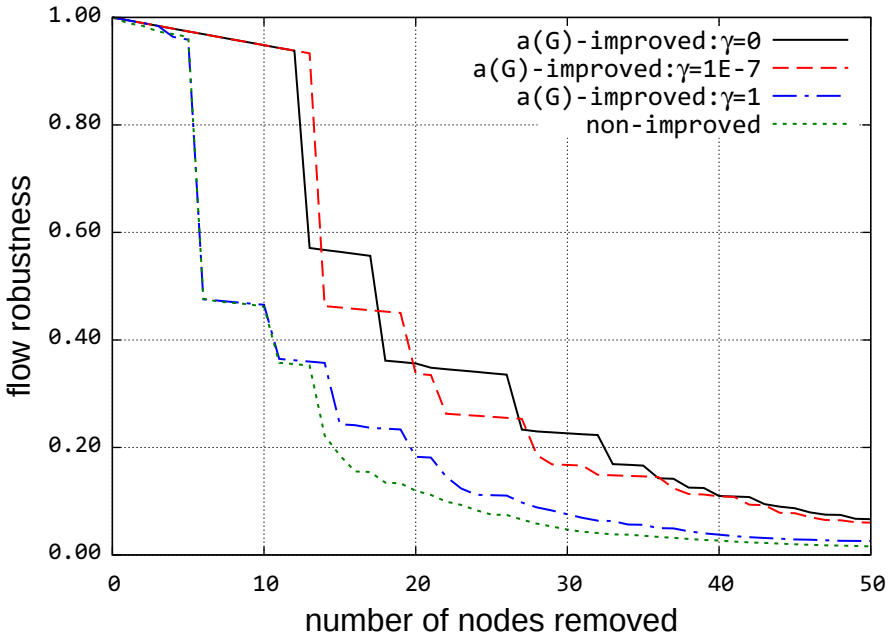

Figure B.16: AT&T betweenness-based attack

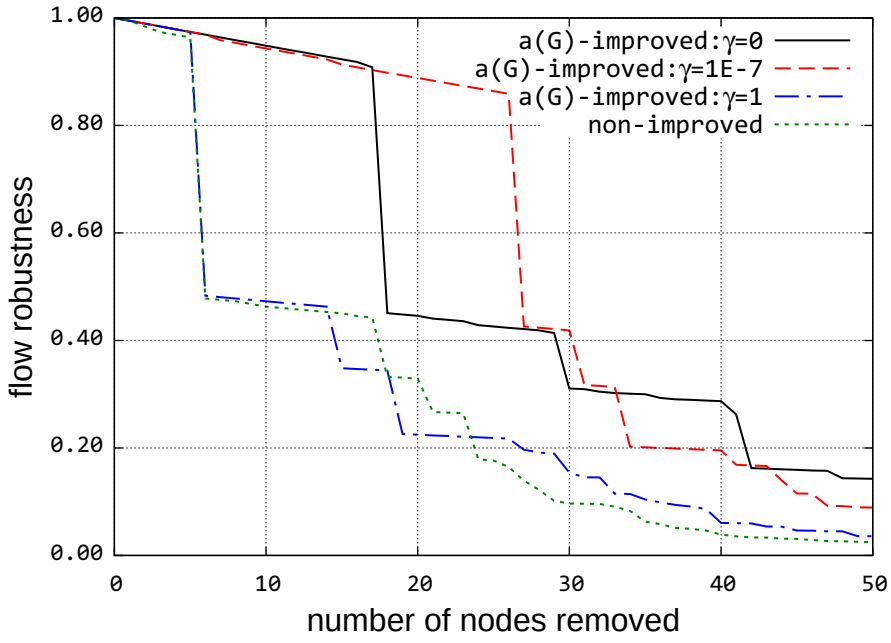

Figure B.17: AT&T closeness-based attack

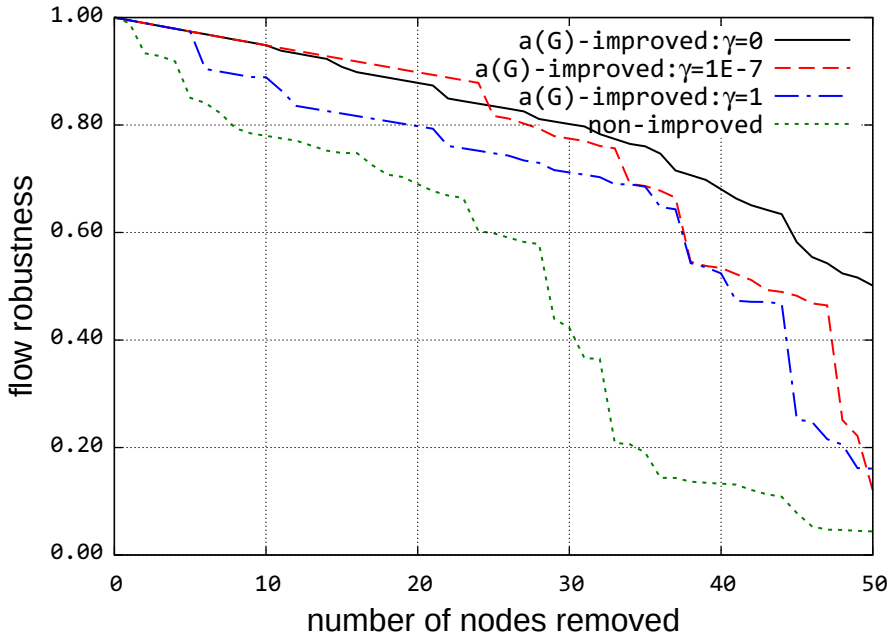

Figure B.18: AT&T degree-based attack

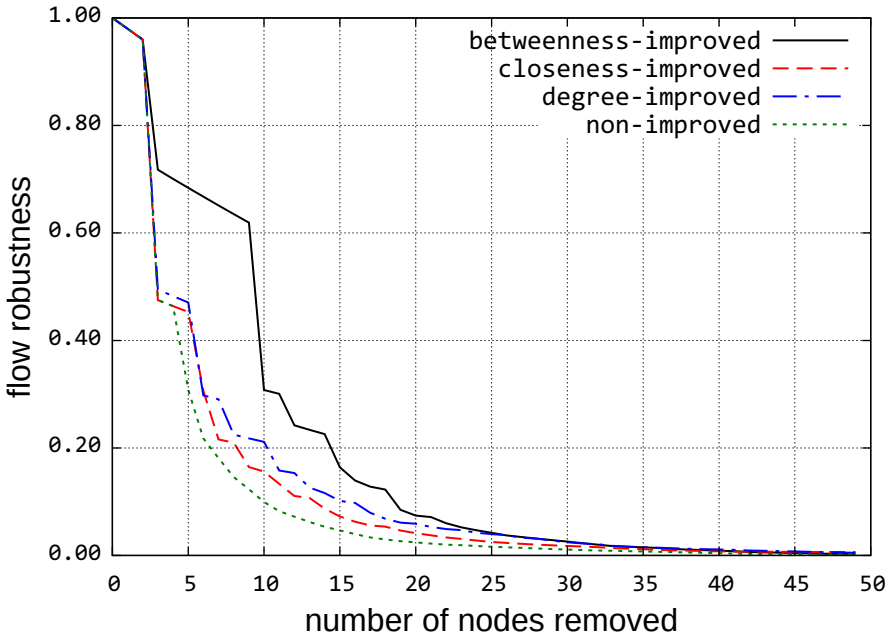

Figure B.19: Level 3 betweenness-based attack

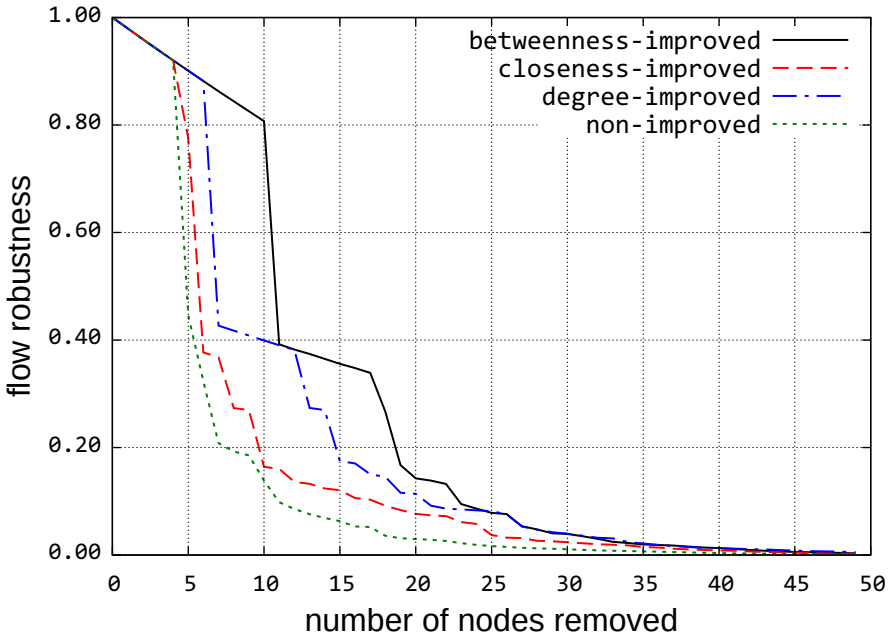

Figure B.20: Level 3 closeness-based attack

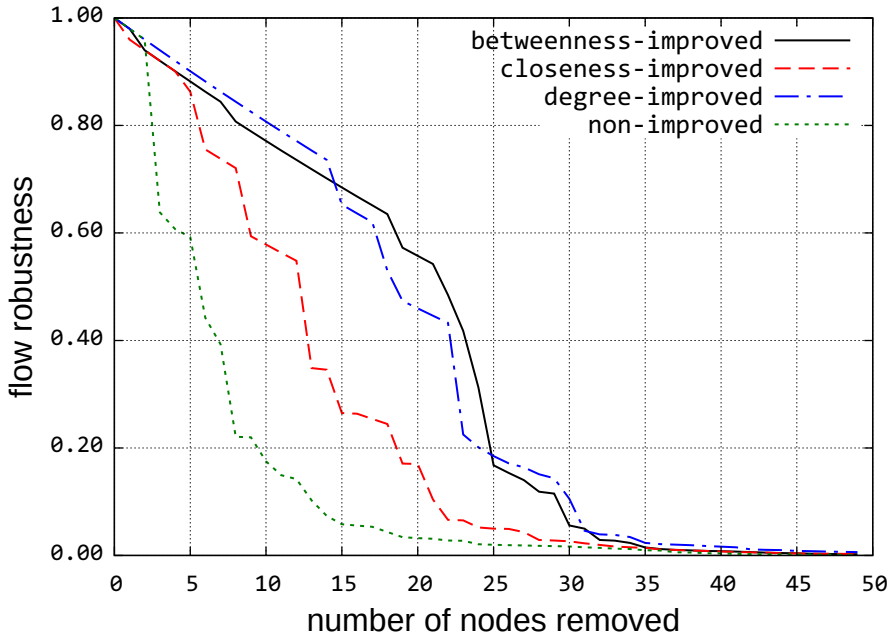

Figure B.21: Level 3 degree-based attack

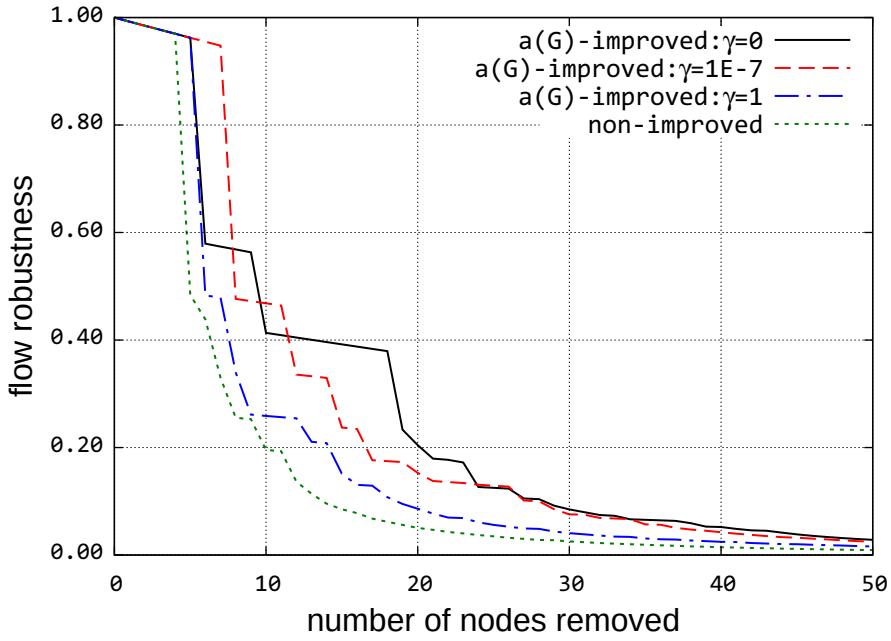

Figure B.22: Sprint betweenness-based attack

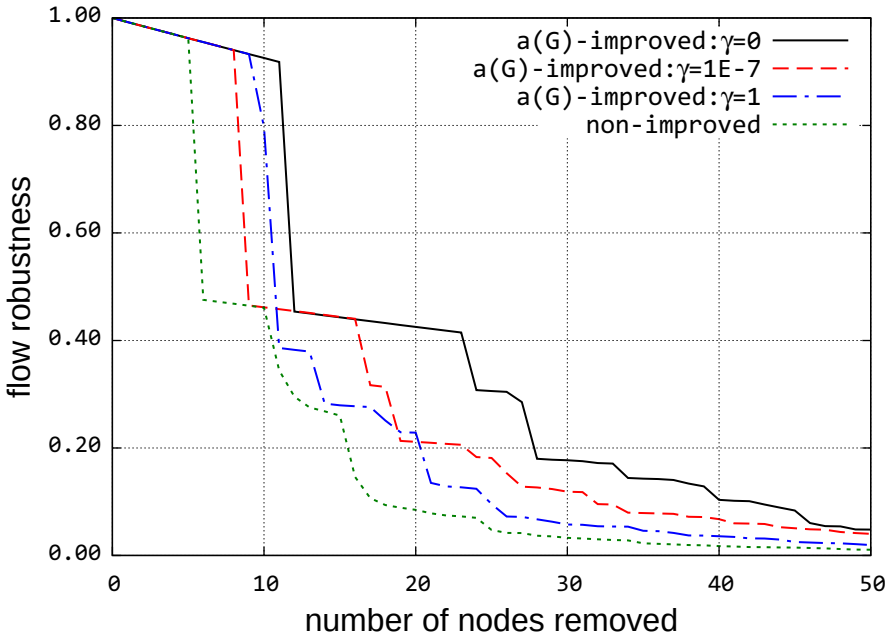

Figure B.23: Sprint closeness-based attack

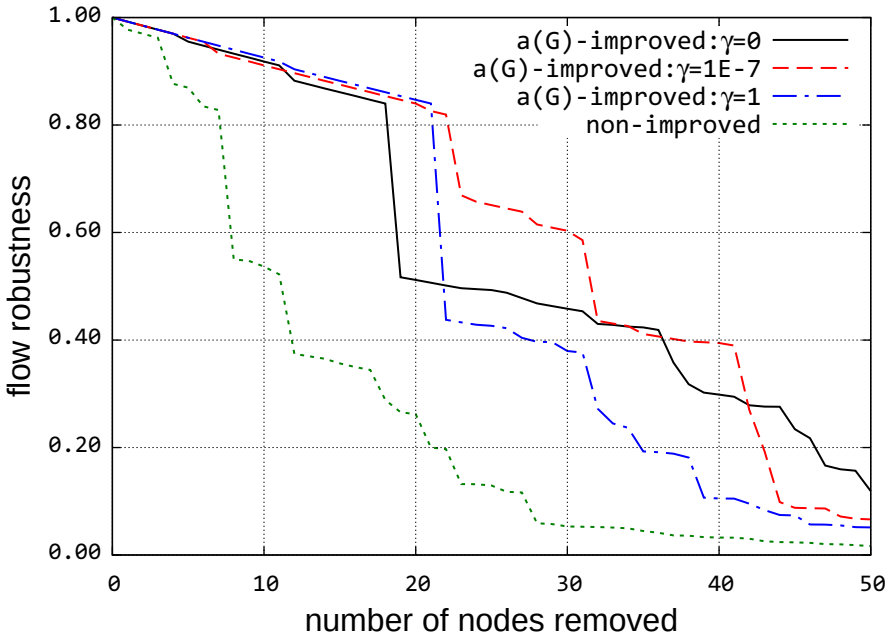

Figure B.24: Sprint degree-based attack

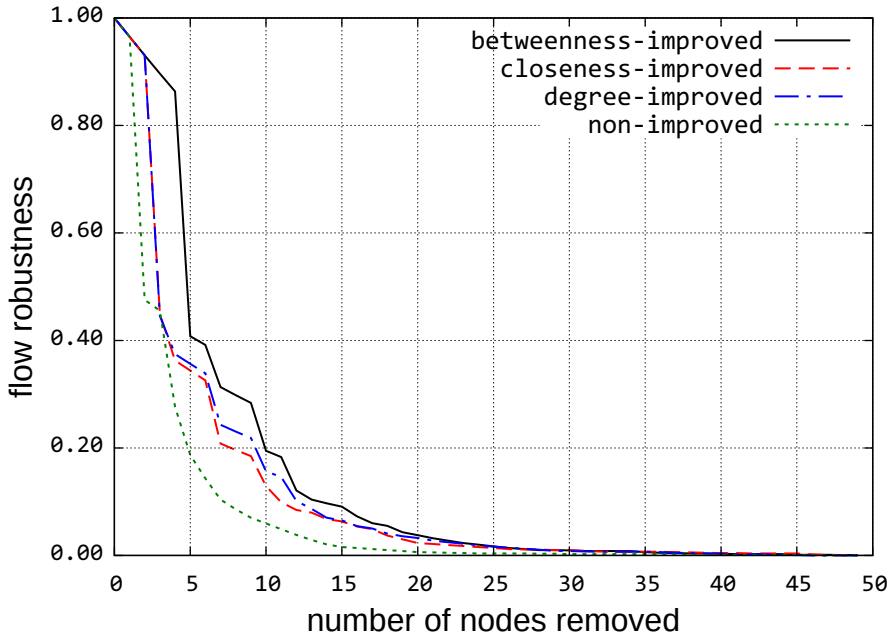

Figure B.25: Internet2 betweenness-based attack

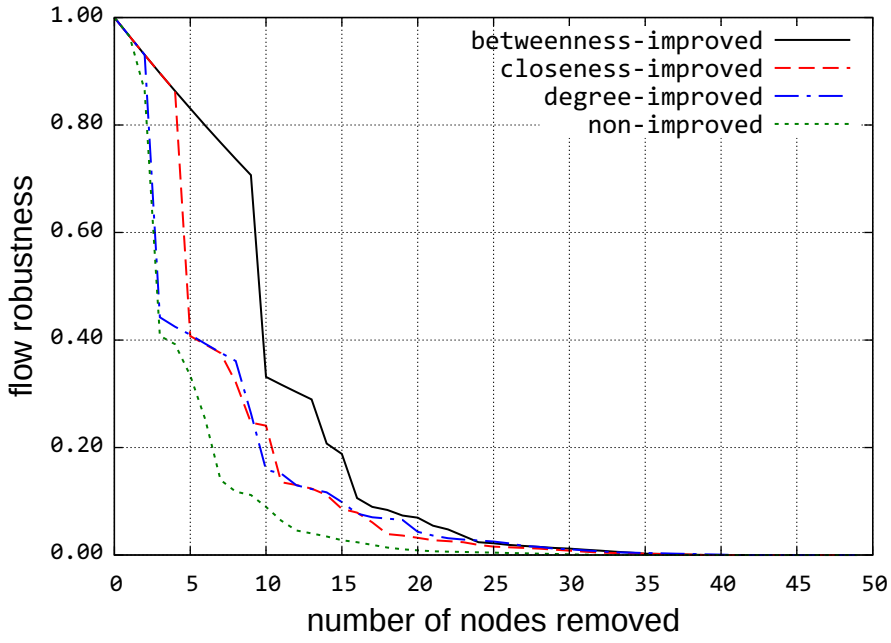

Figure B.26: Internet2 closeness-based attack

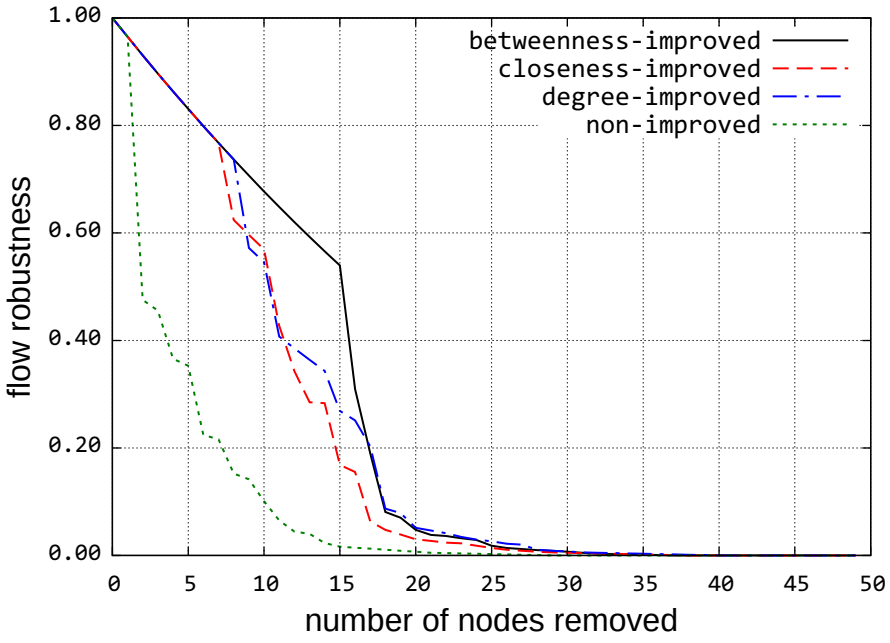

Figure B.27: Internet2 degree-based attack

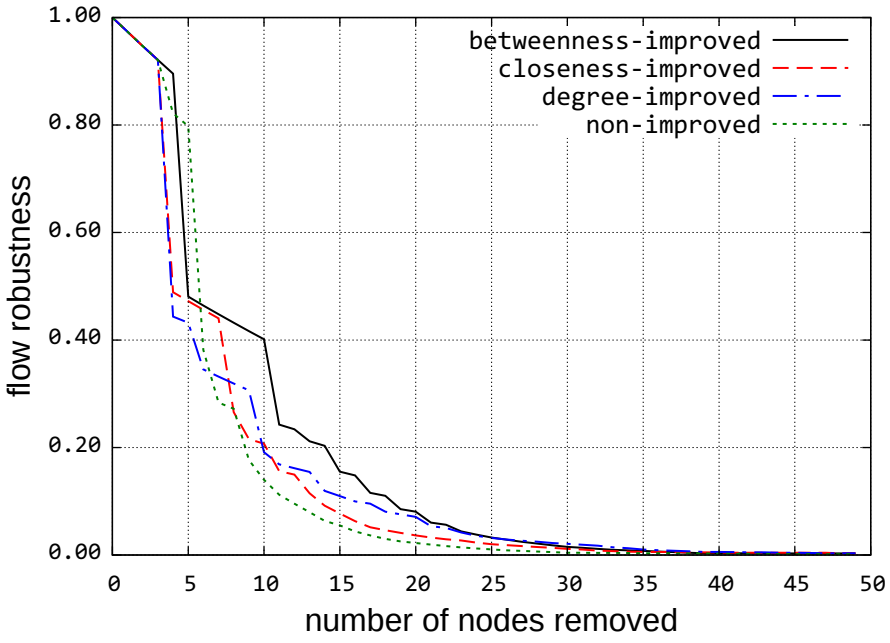

Figure B.28: CORONET betweenness-based attack

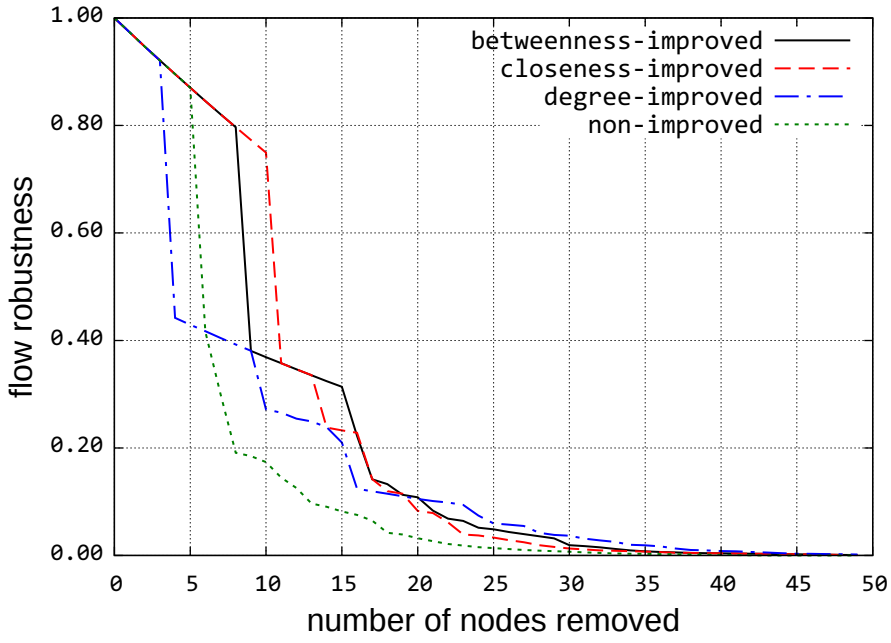

Figure B.29: CORONET closeness-based attack

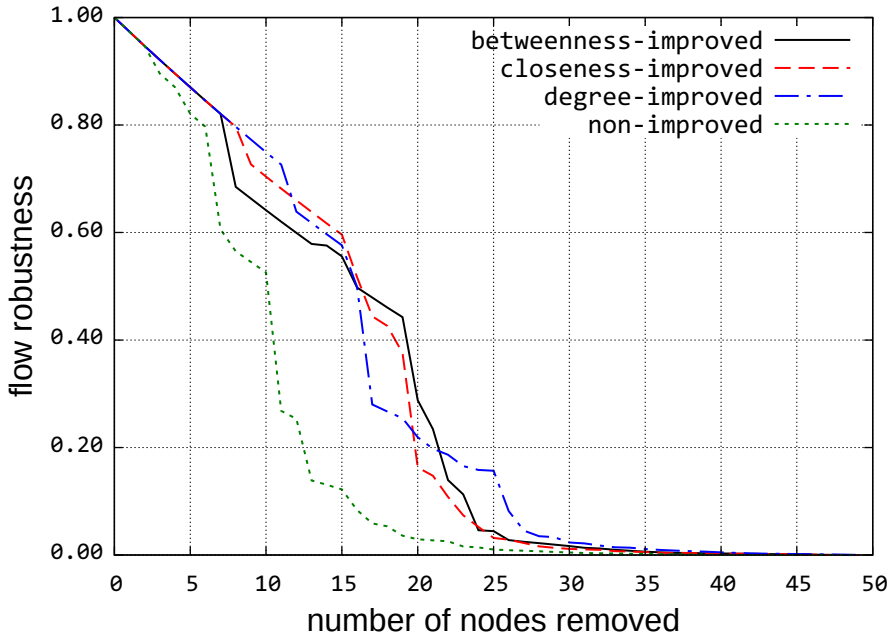

Figure B.30: CORONET degree-based attack

B.2 Graph Improvement via Path Diversity

B.2.1 Impact of Varying Hop Count Threshold on TGD and Cost

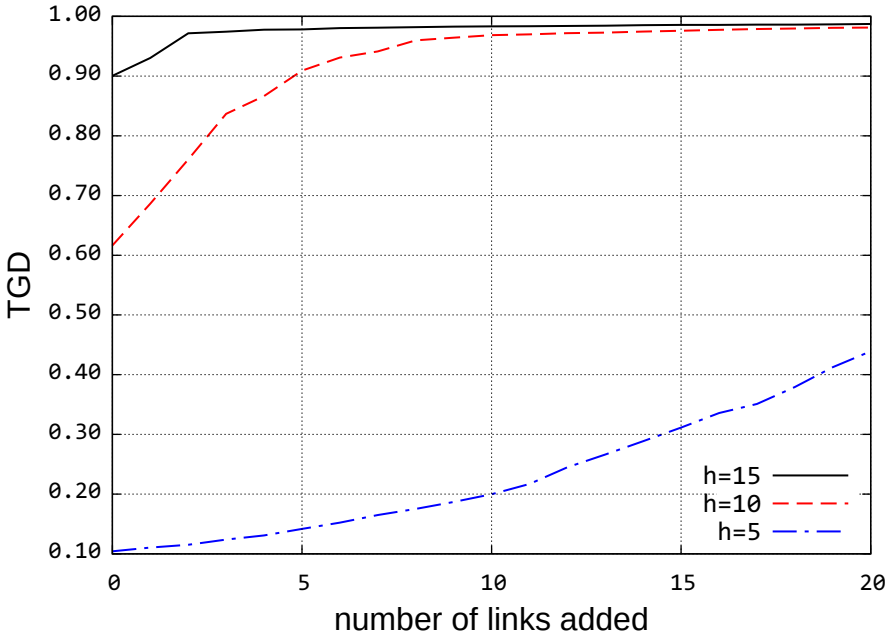

Figure B.31: CORONET TGD improvement

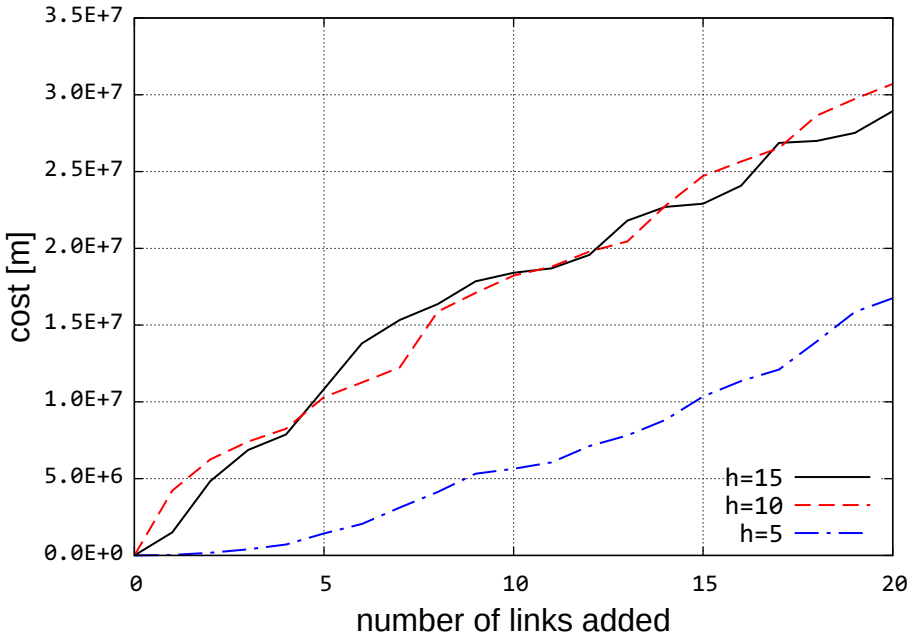

Figure B.32: CORONET cost incurred

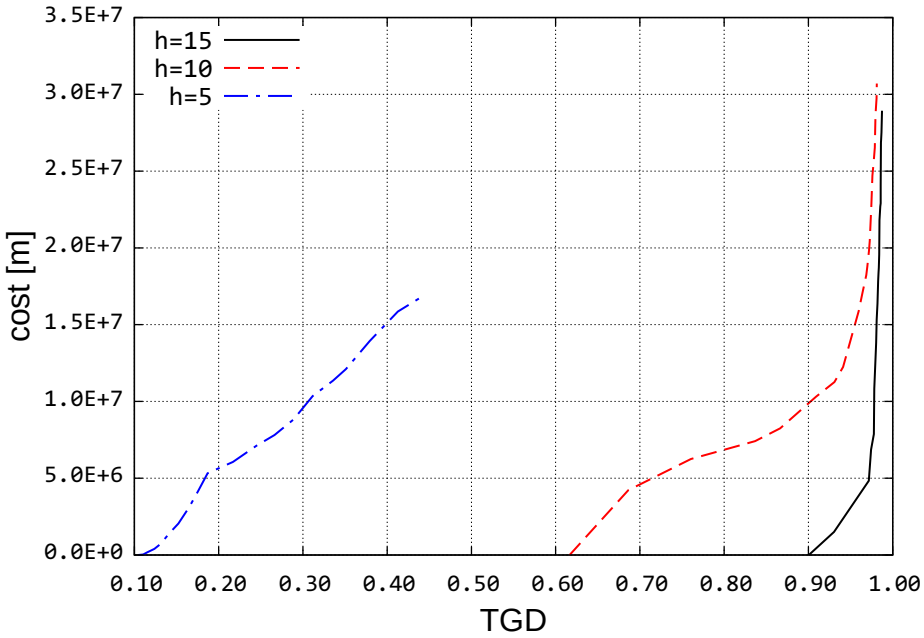

Figure B.33: CORONET cost and TGD

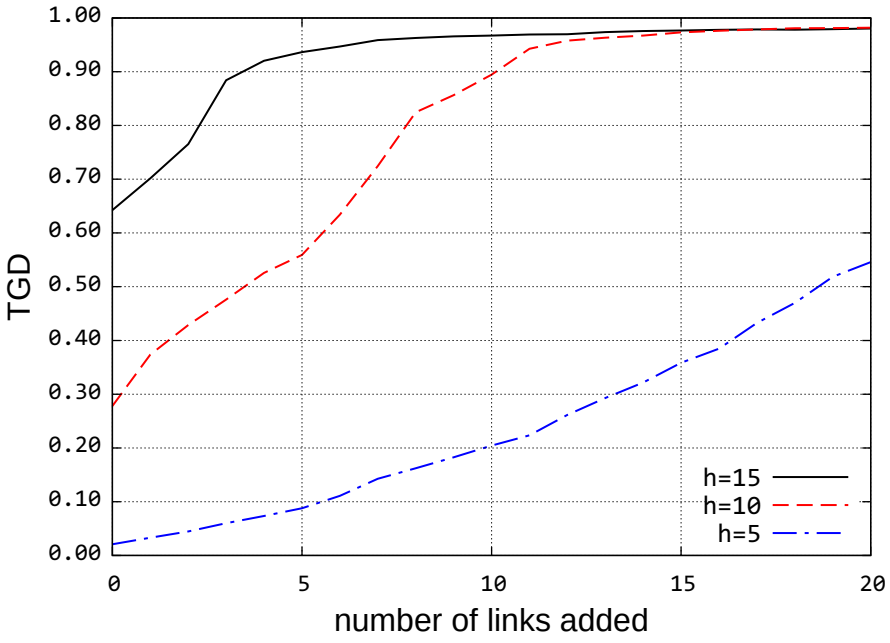

Figure B.34: Internet2 TGD improvement

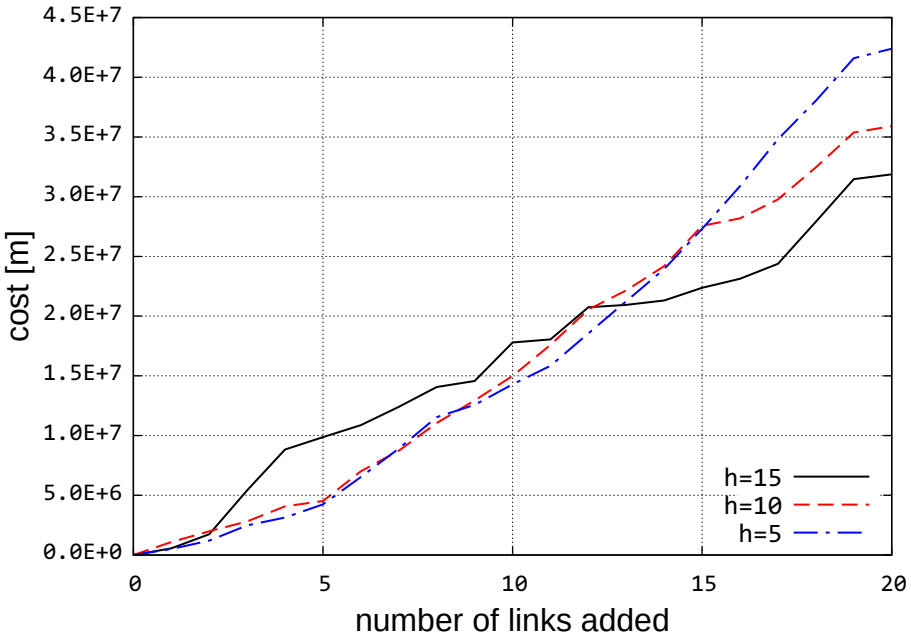

Figure B.35: Internet2 cost incurred

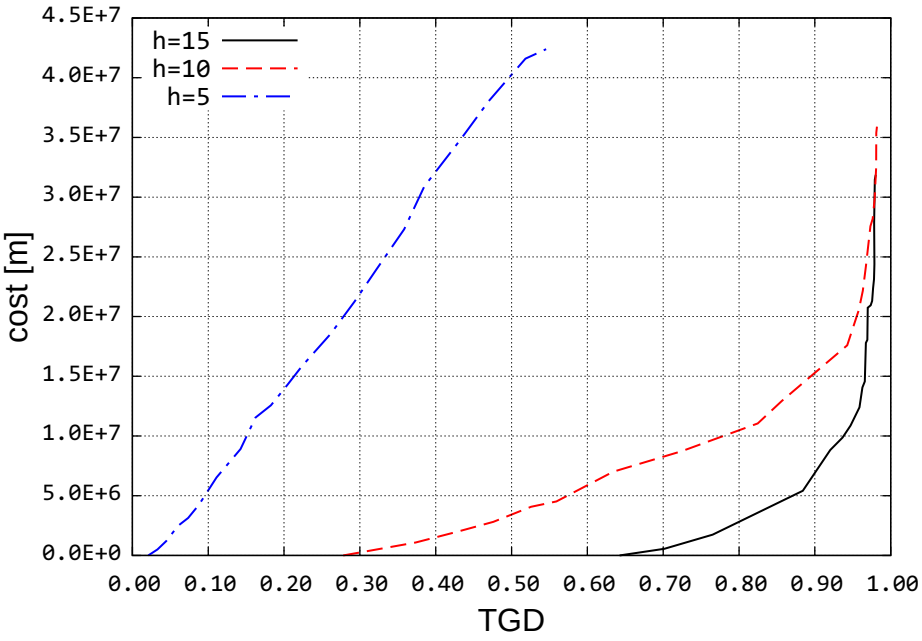

Figure B.36: Internet2 cost and TGD

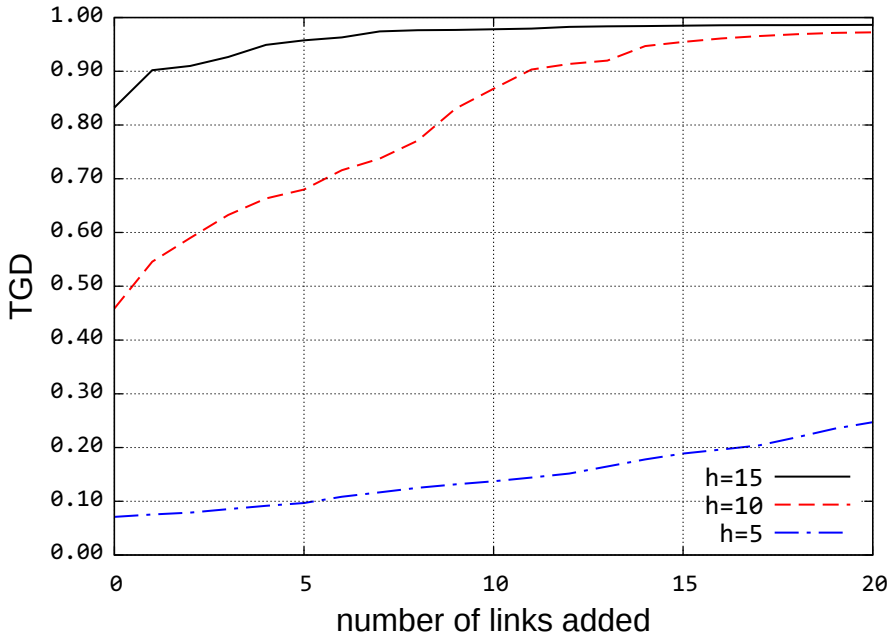

Figure B.37: Level 3 TGD improvement

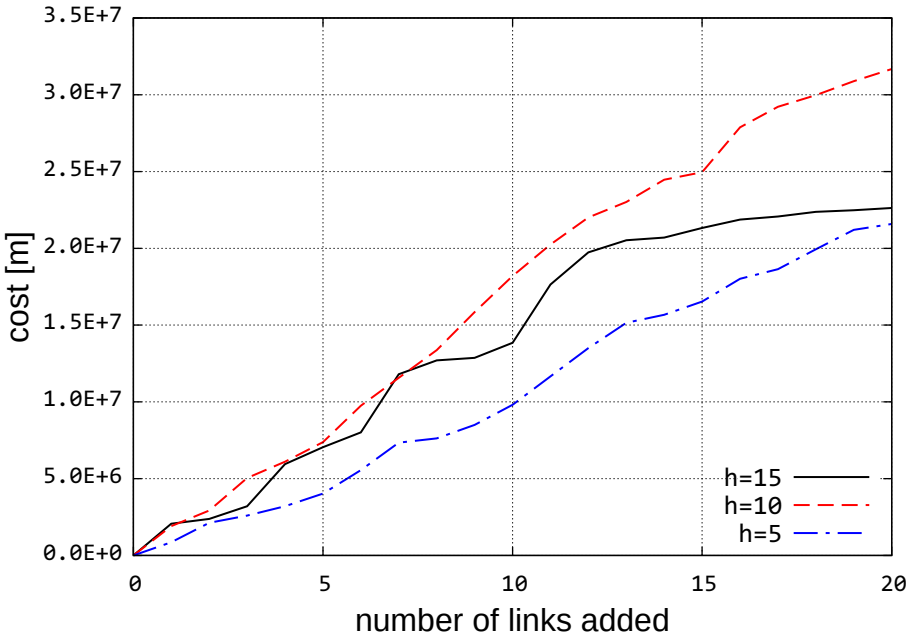

Figure B.38: Level 3 cost incurred

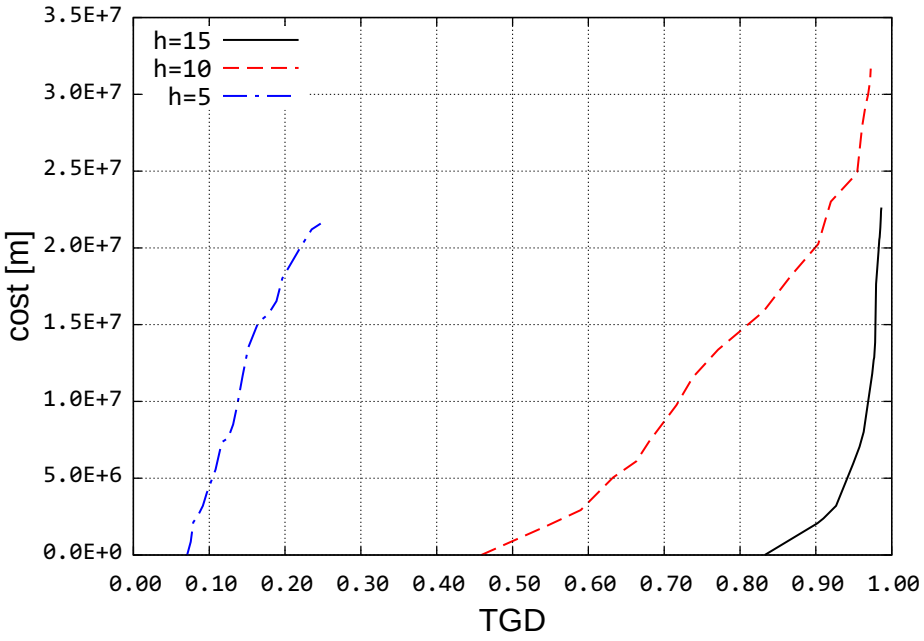

Figure B.39: Level 3 cost and TGD

B.2.2 Impact of Varying k on TGD and Cost

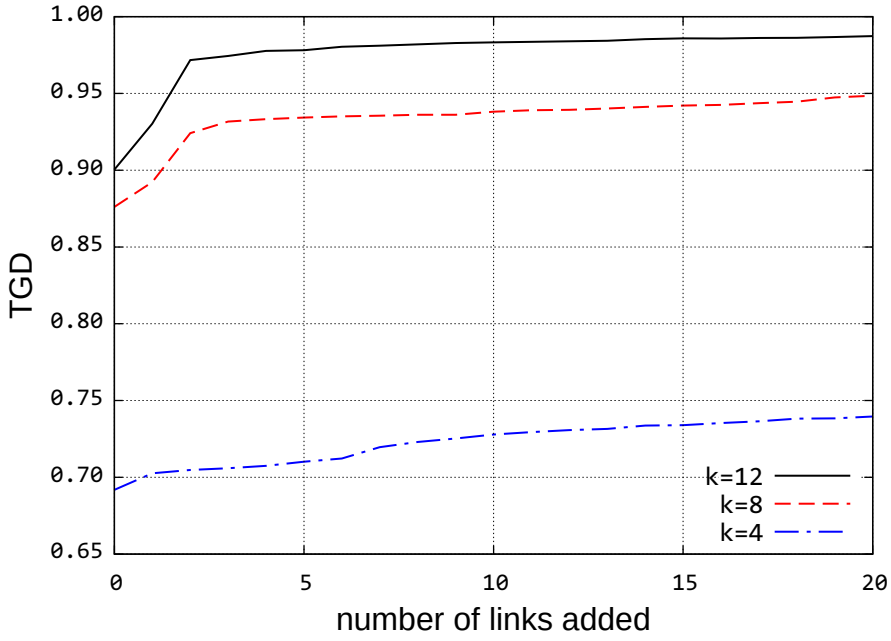

Figure B.40: CORONET TGD improvement

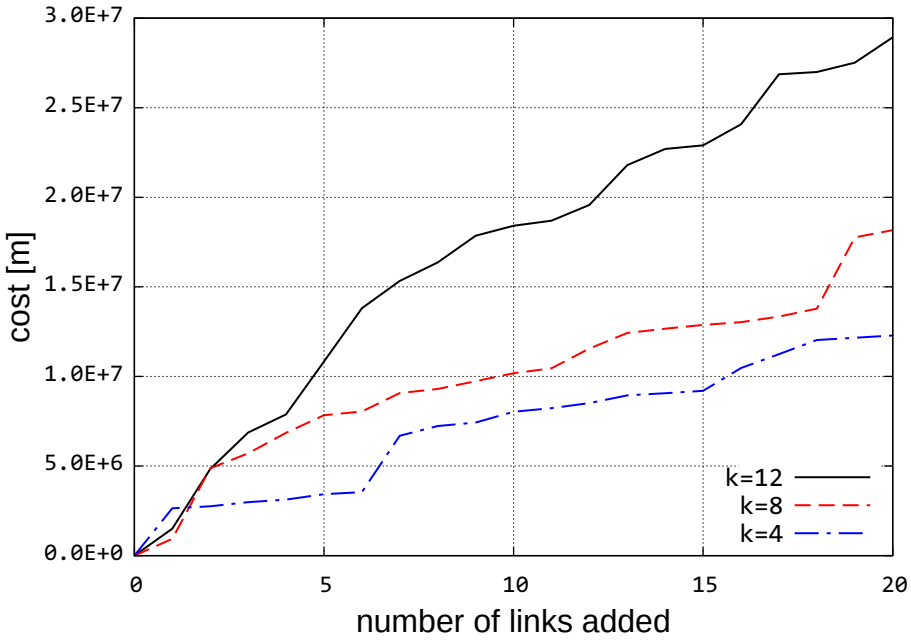

Figure B.41: CORONET cost incurred

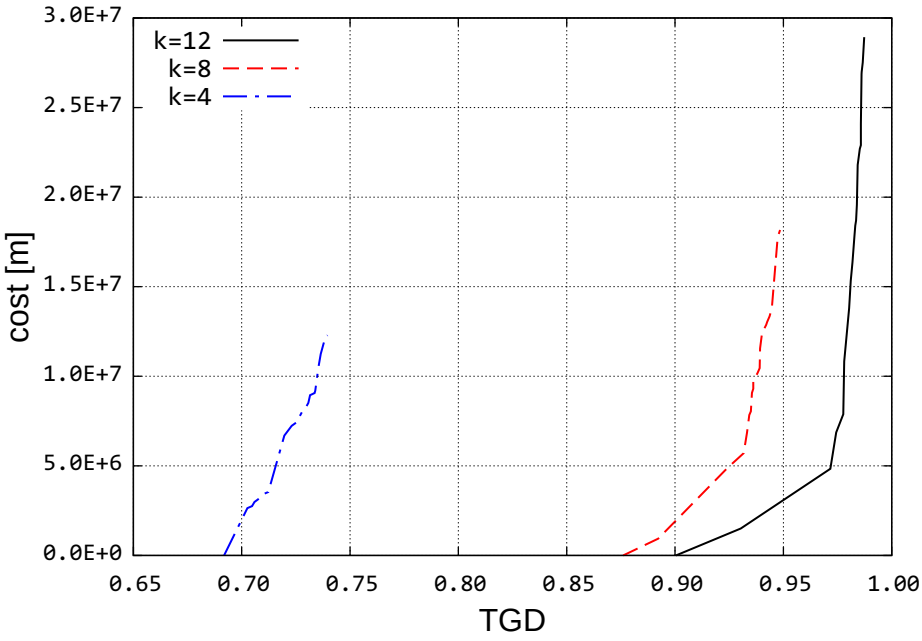

Figure B.42: CORONET cost and TGD

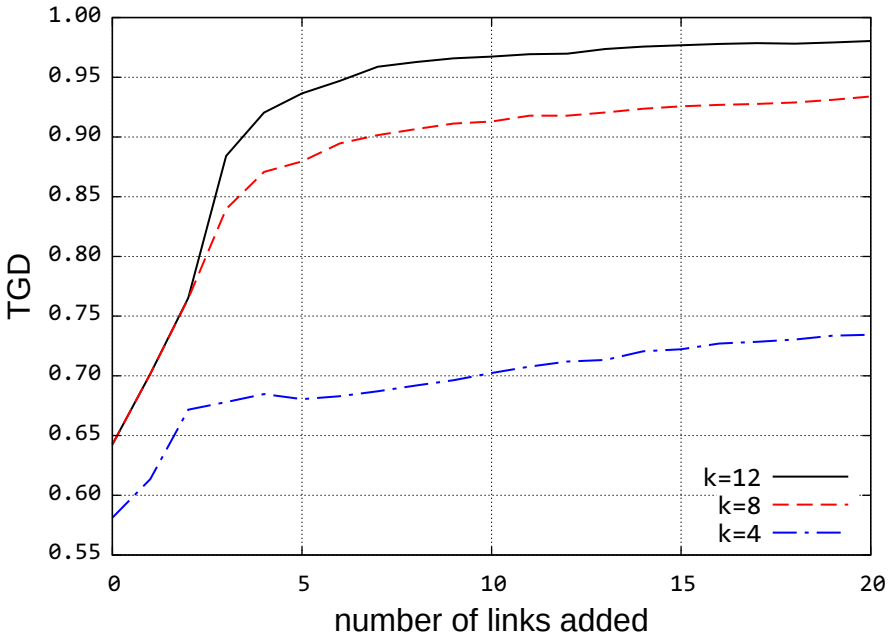

Figure B.43: Internet2 TGD improvement

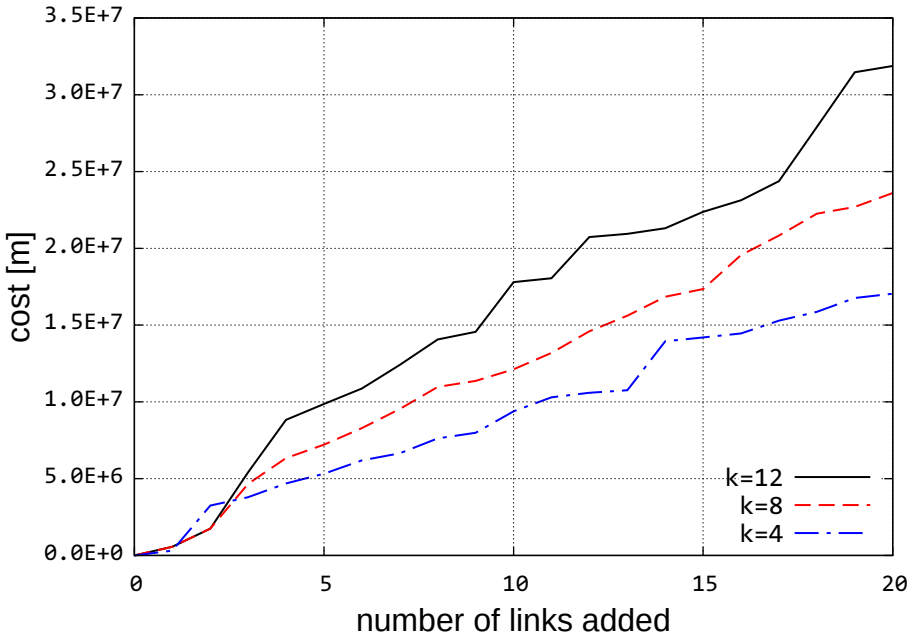

Figure B.44: Internet2 cost incurred

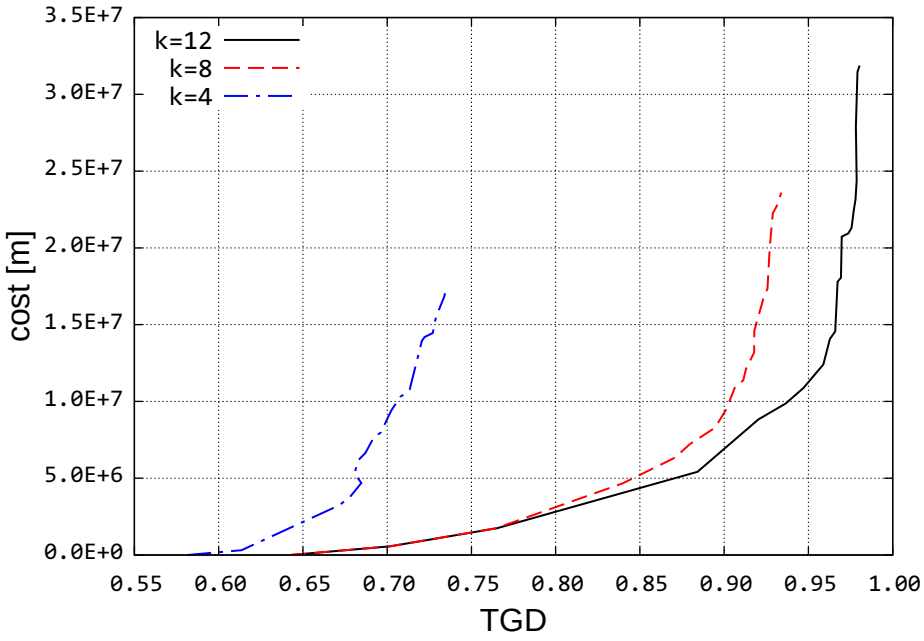

Figure B.45: Internet2 cost and TGD

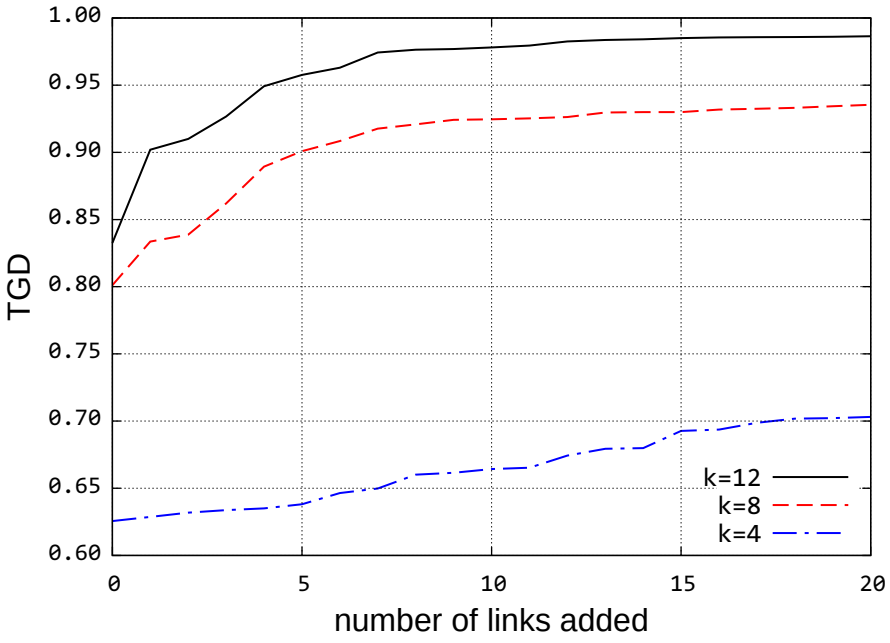

Figure B.46: Level 3 TGD improvement

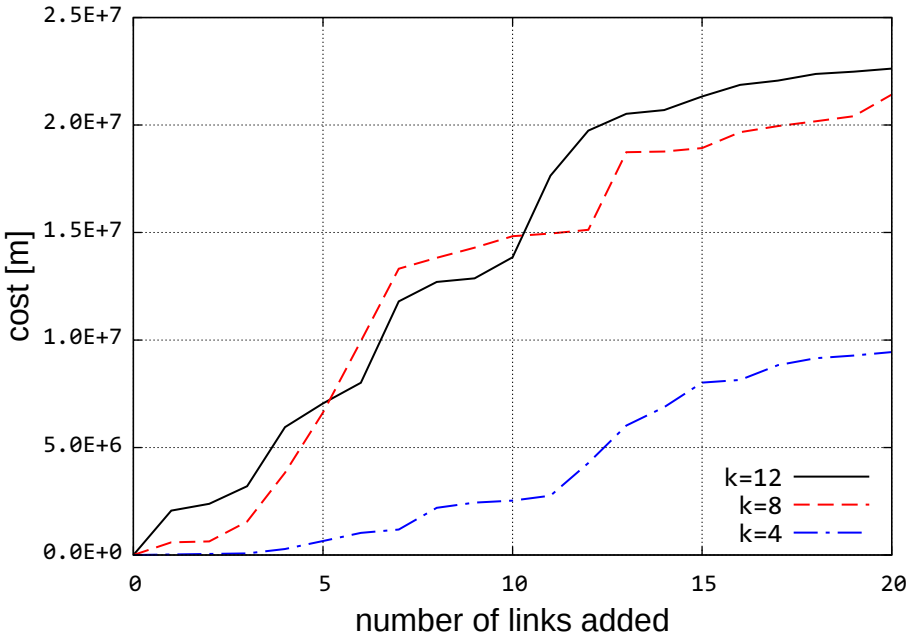

Figure B.47: Level 3 cost incurred

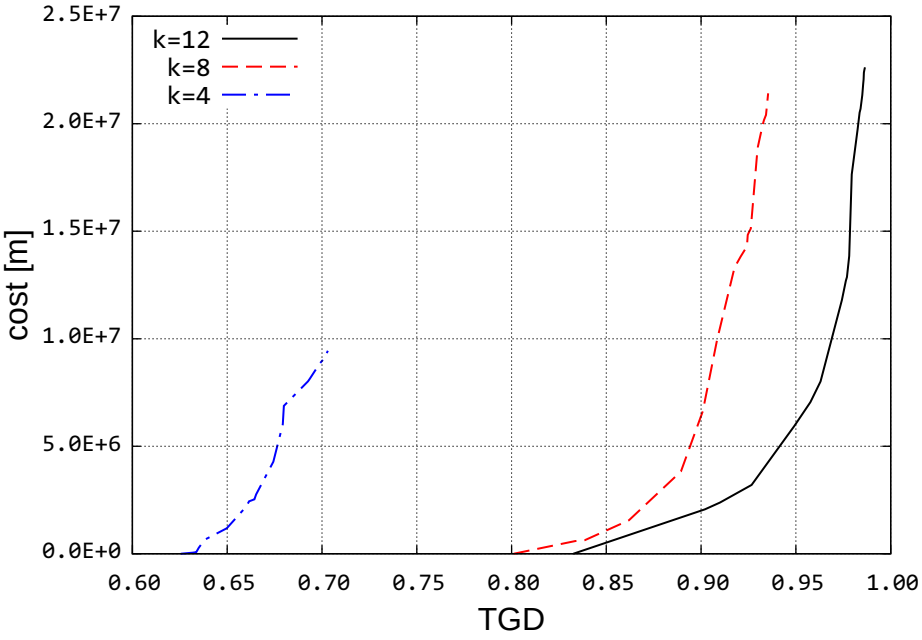

Figure B.48: Level 3 cost and TGD

B.2.3 Flow Robustness Evaluation of PD-improved Graphs

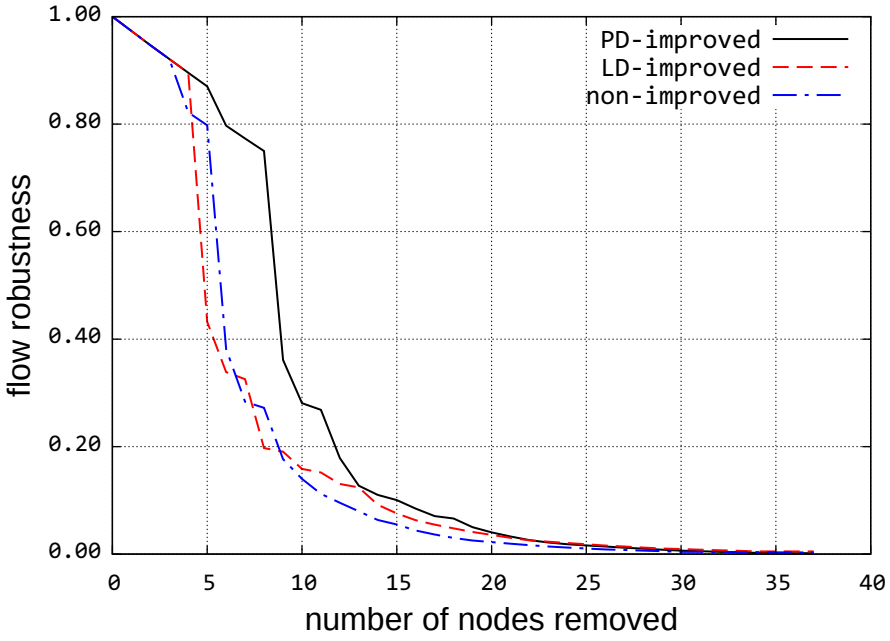

Figure B.49: CORONET betweenness-based attack

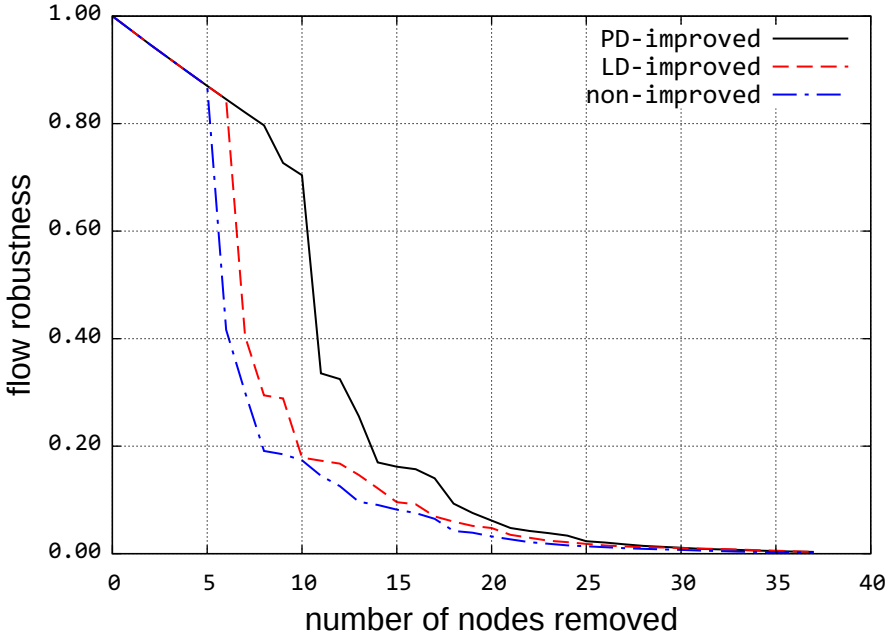

Figure B.50: CORONET closeness-based attack

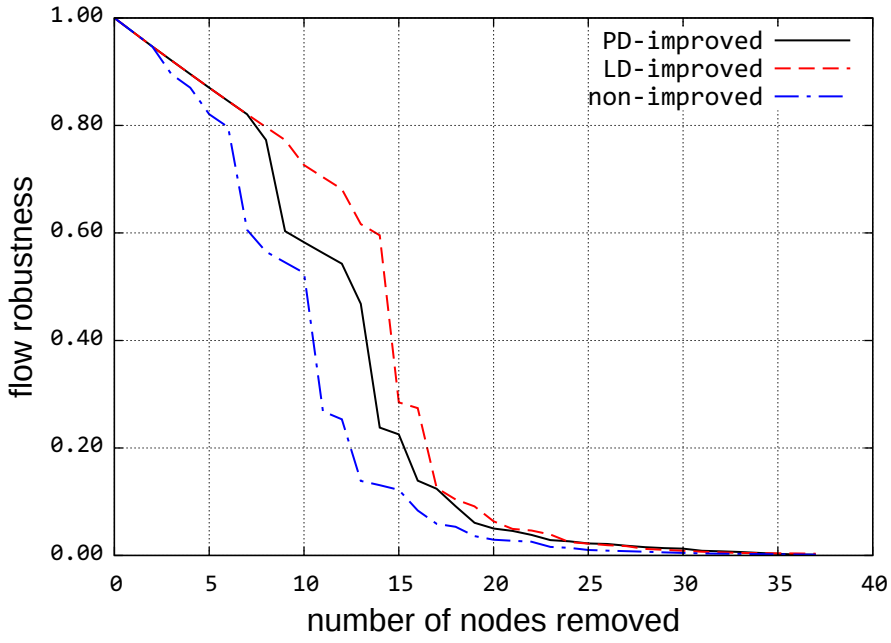

Figure B.51: CORONET degree-based attack

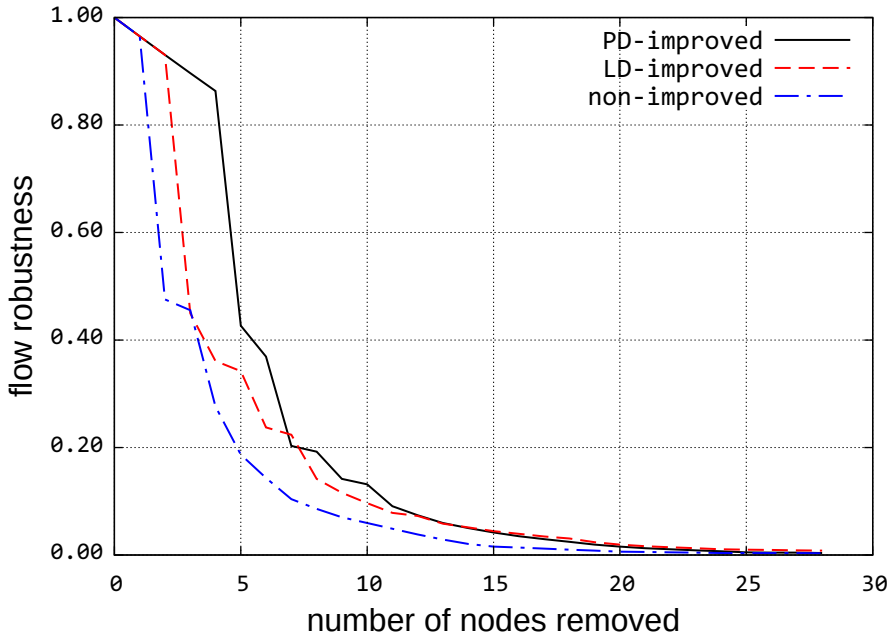

Figure B.52: Internet2 betweenness-based attack

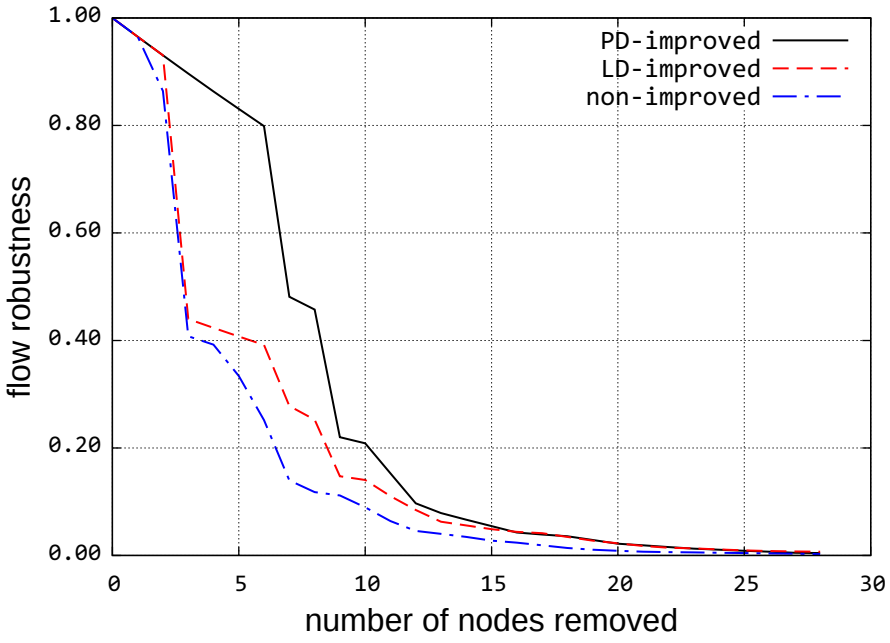

Figure B.53: Internet2 closeness-based attack

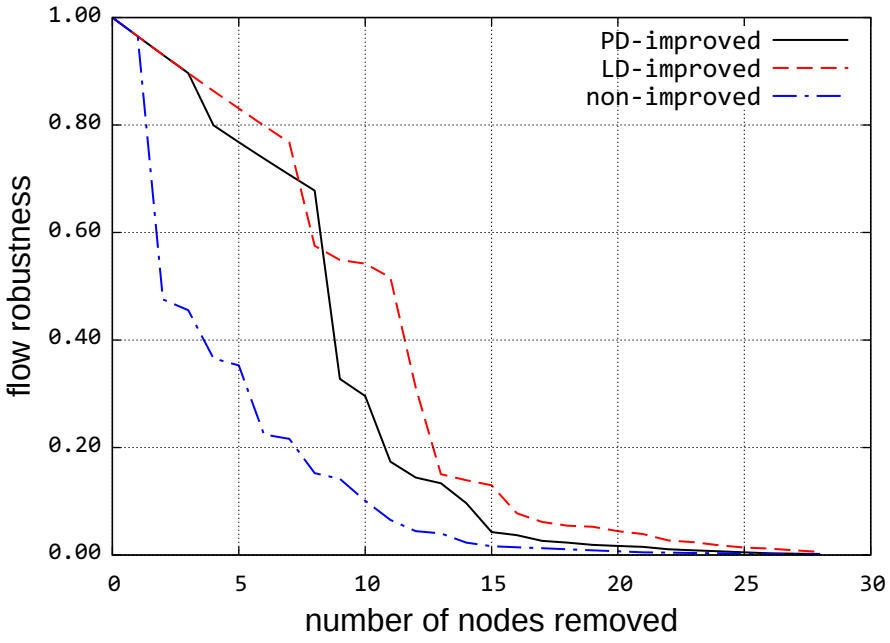

Figure B.54: Internet2 degree-based attack

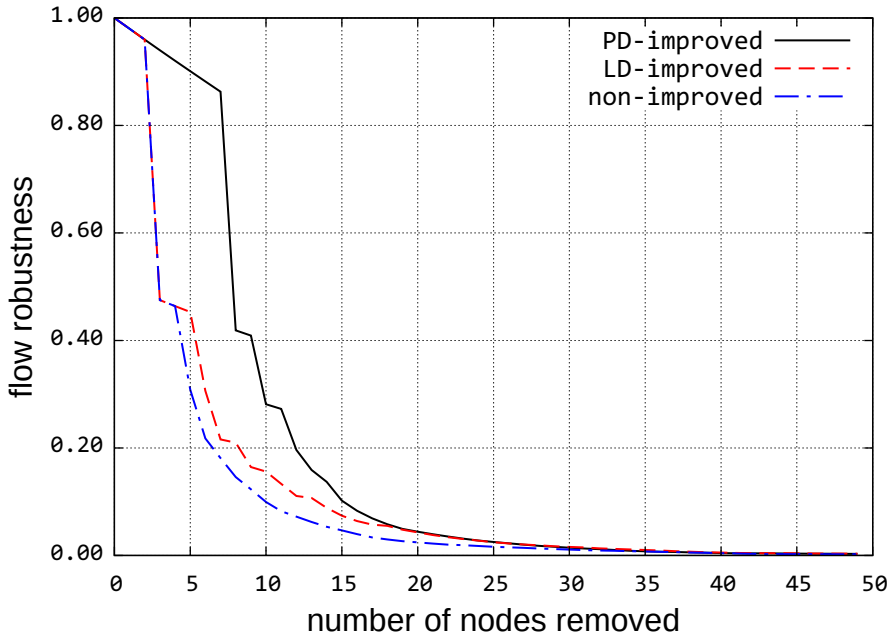

Figure B.55: Level 3 betweenness-based attack

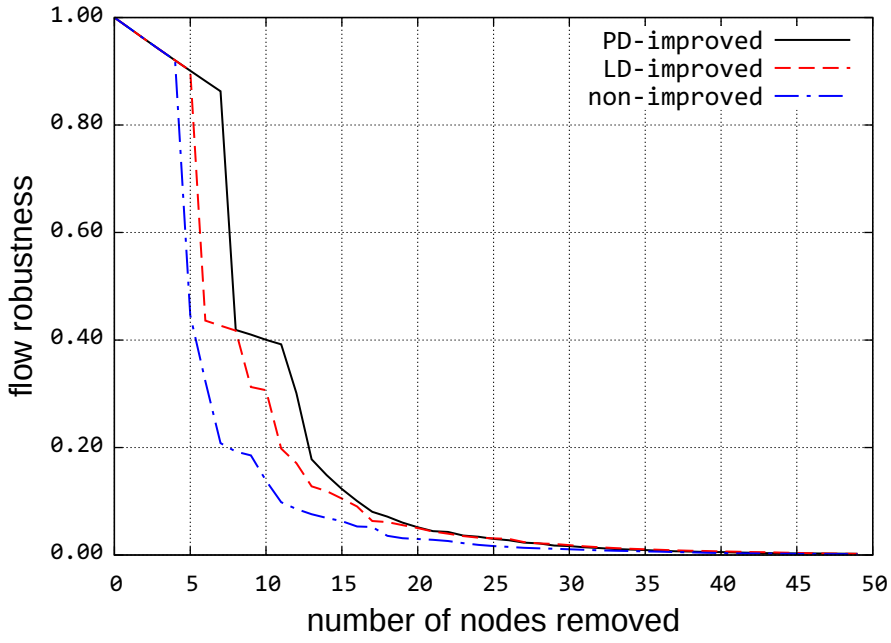

Figure B.56: Level 3 closeness-based attack

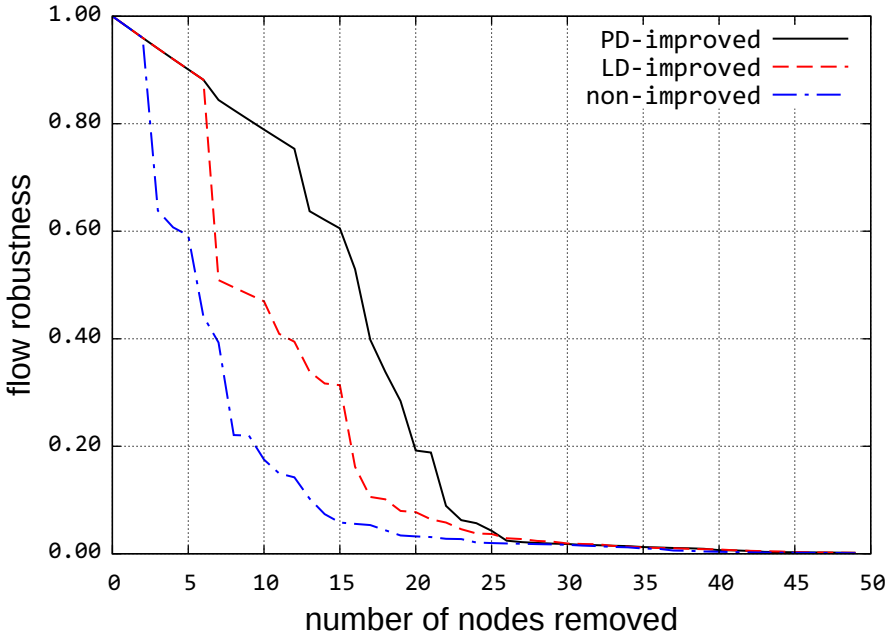

Figure B.57: Level 3 degree-based attack

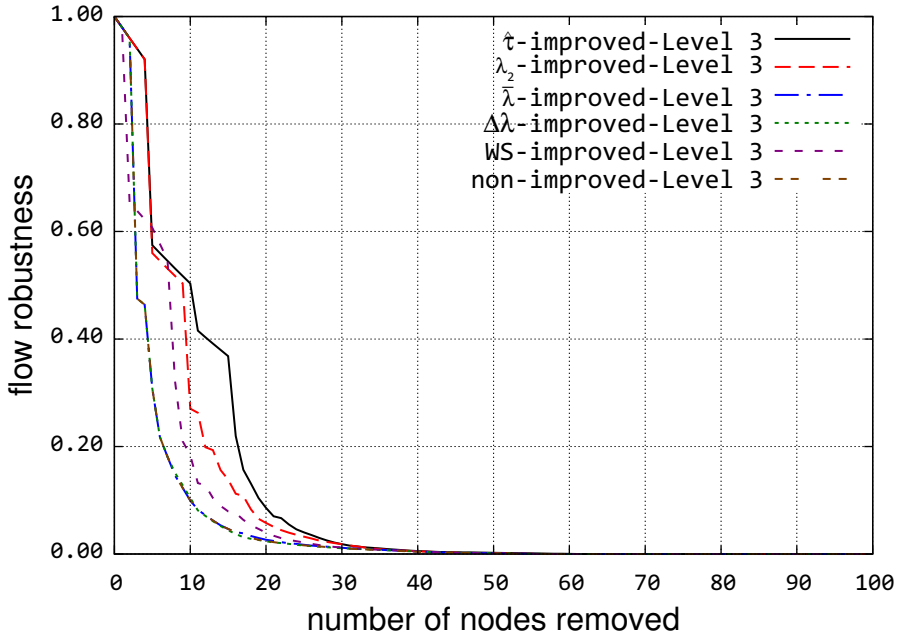

Figure B.58: Level 3 betweenness-based attack

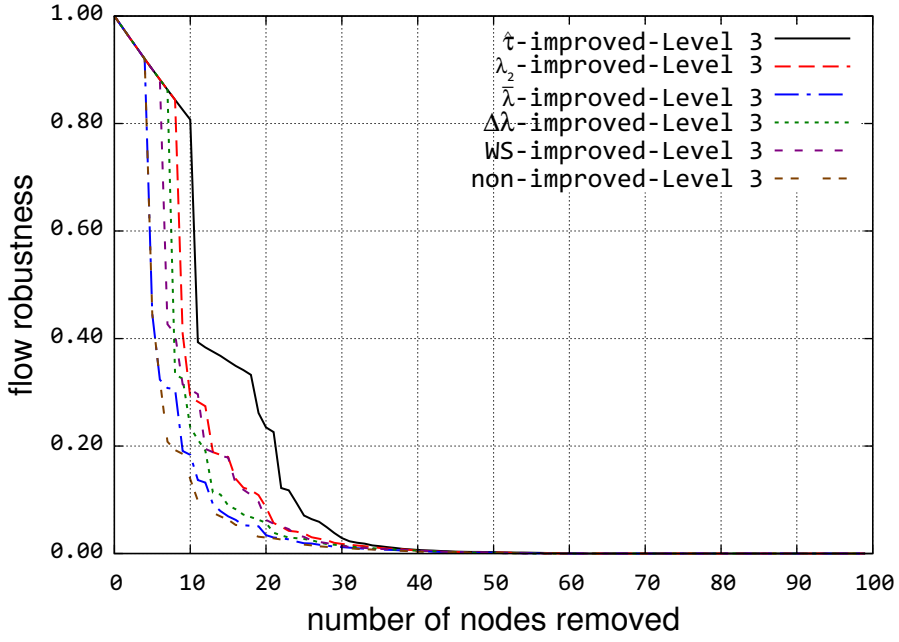

Figure B.59: Level 3 closeness-based attack

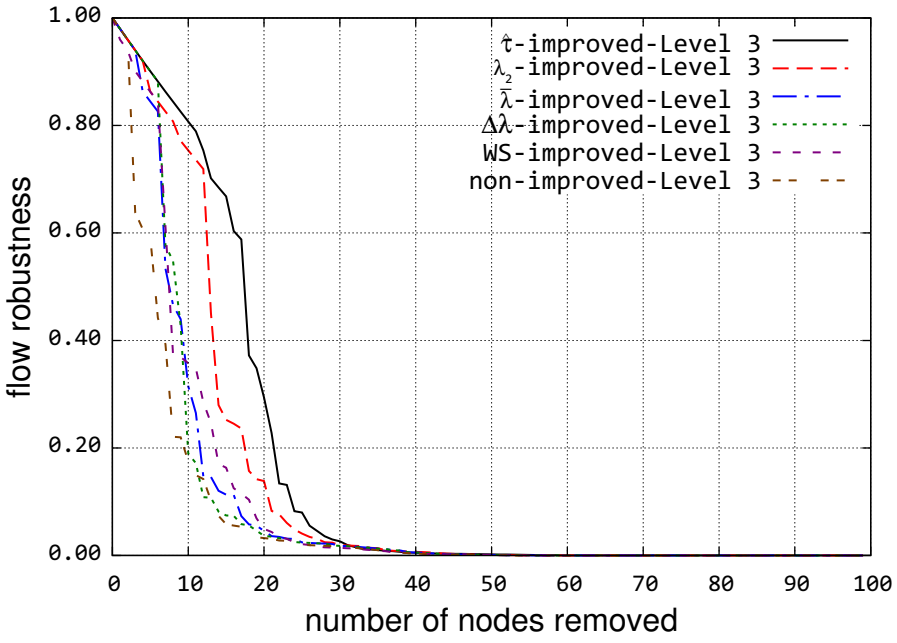

Figure B.60: Level 3 degree-based attack

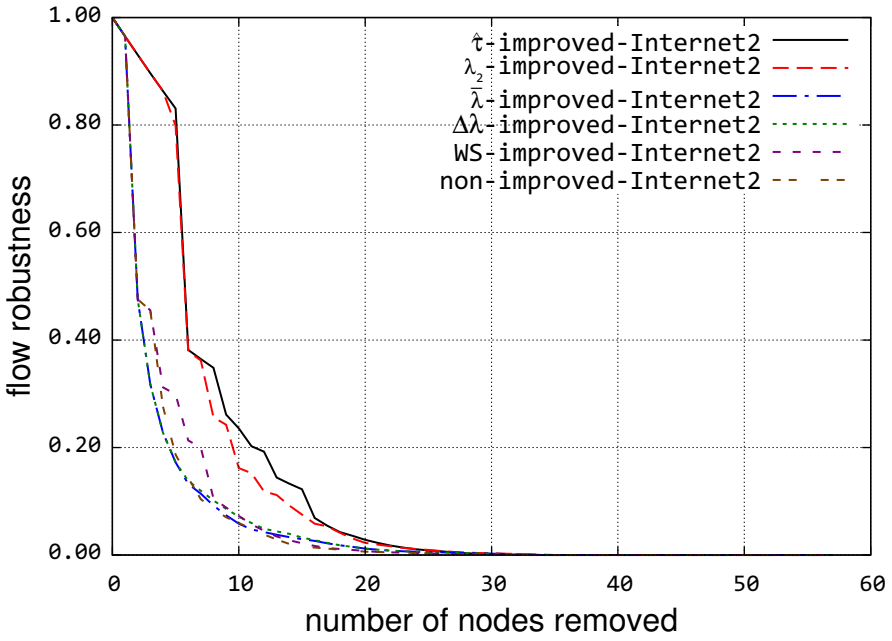

Figure B.61: Internet2 betweenness-based attack

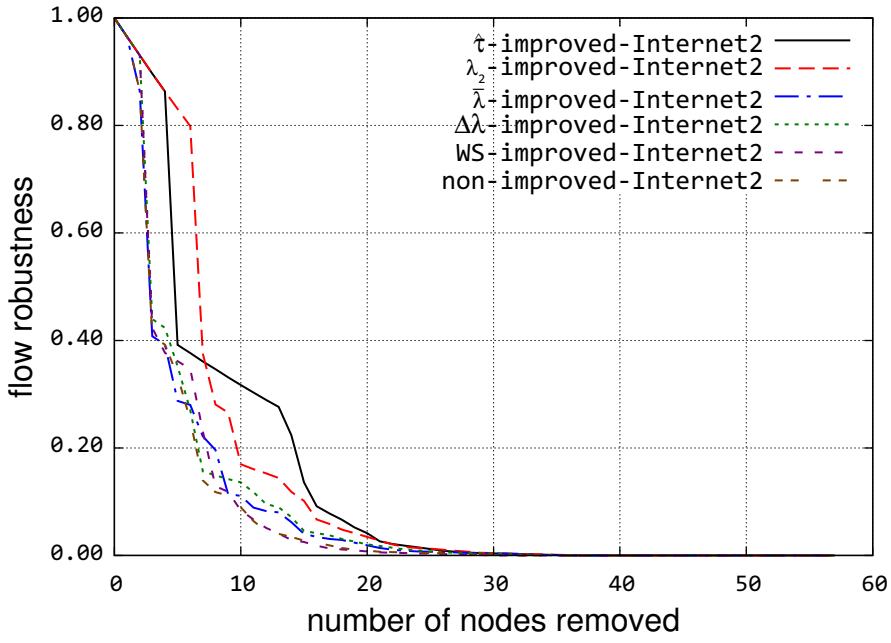

Figure B.62: Internet2 closeness-based attack

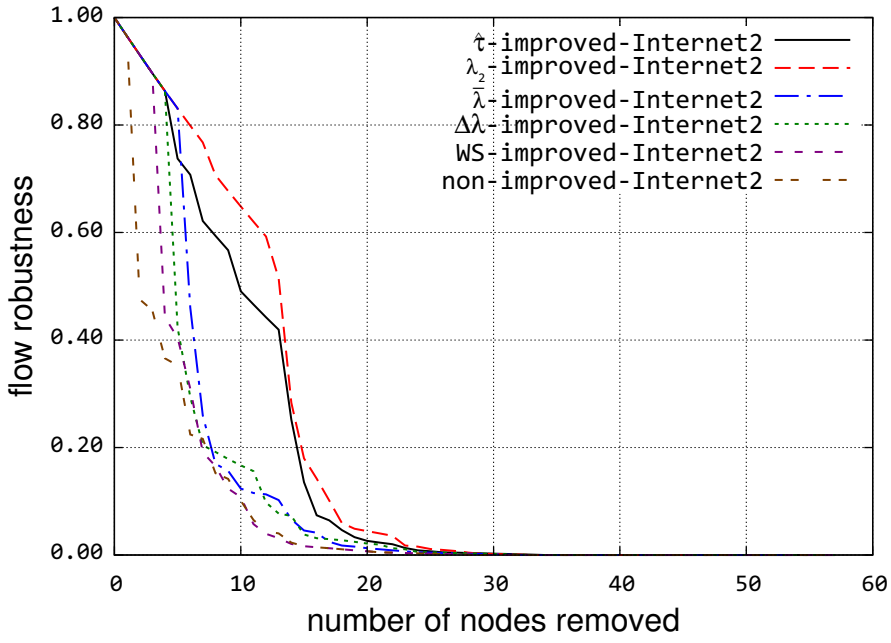

Figure B.63: Internet2 degree-based attack

B.3 Spectral Graph Robustness Improvement

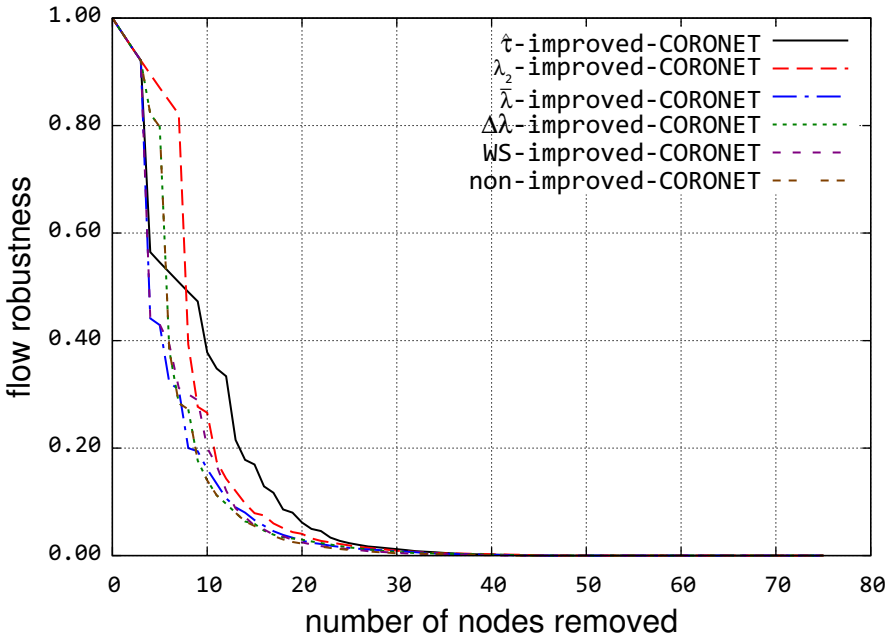

Figure B.64: CORONET betweenness-based attack

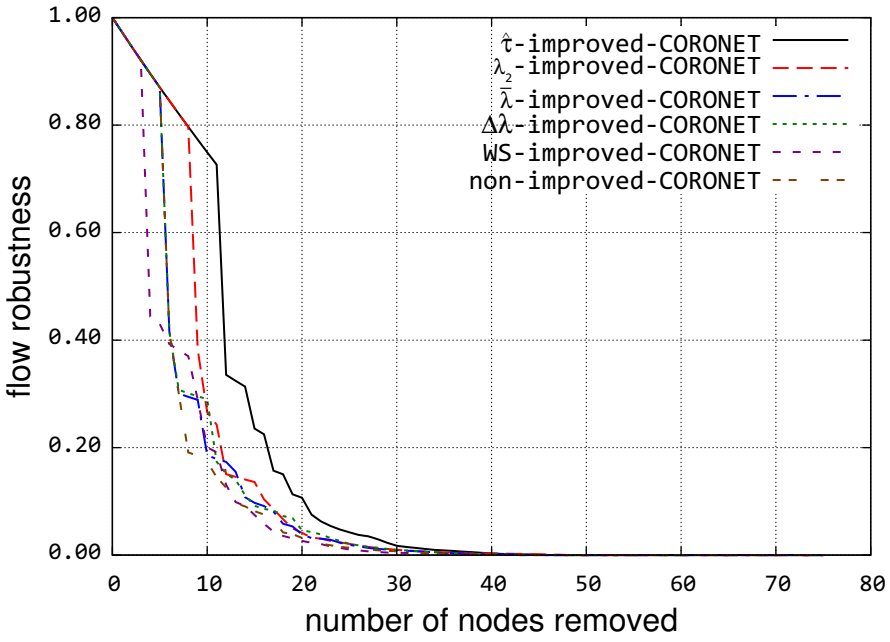

Figure B.65: CORONET closeness-based attack

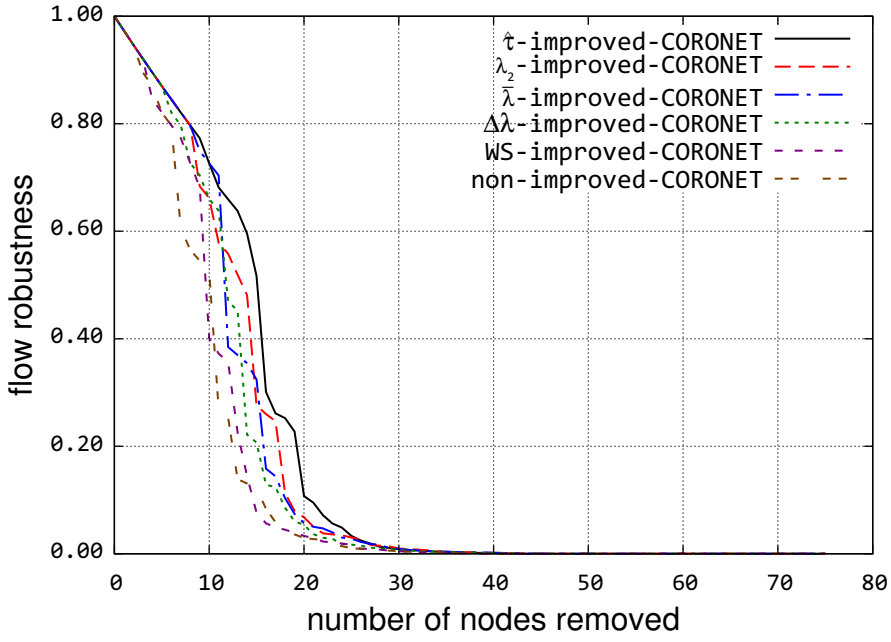

Figure B.66: CORONET degree-based attack

B.4 Unweighted Graphs Improvement Evaluation

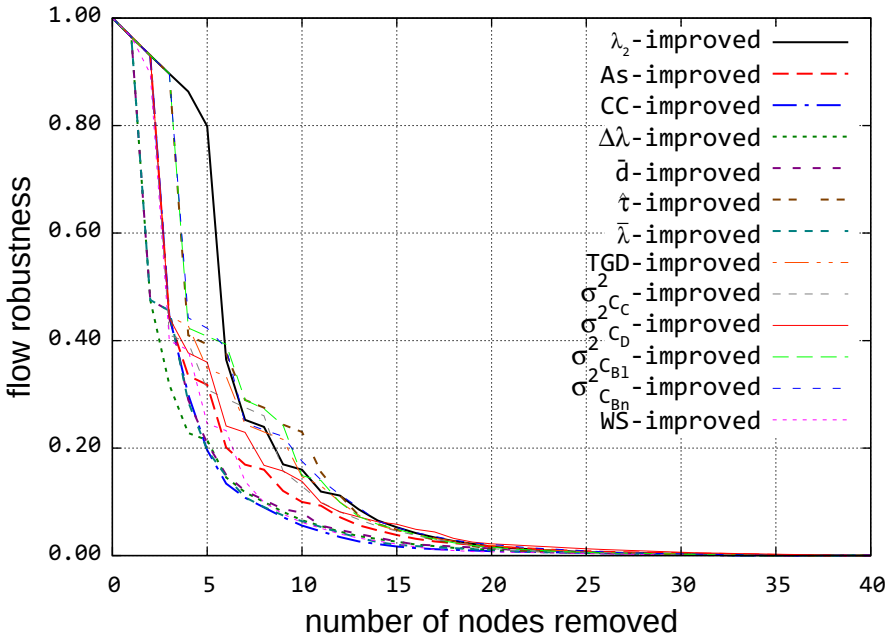

Figure B.67: Internet2 betweenness attack

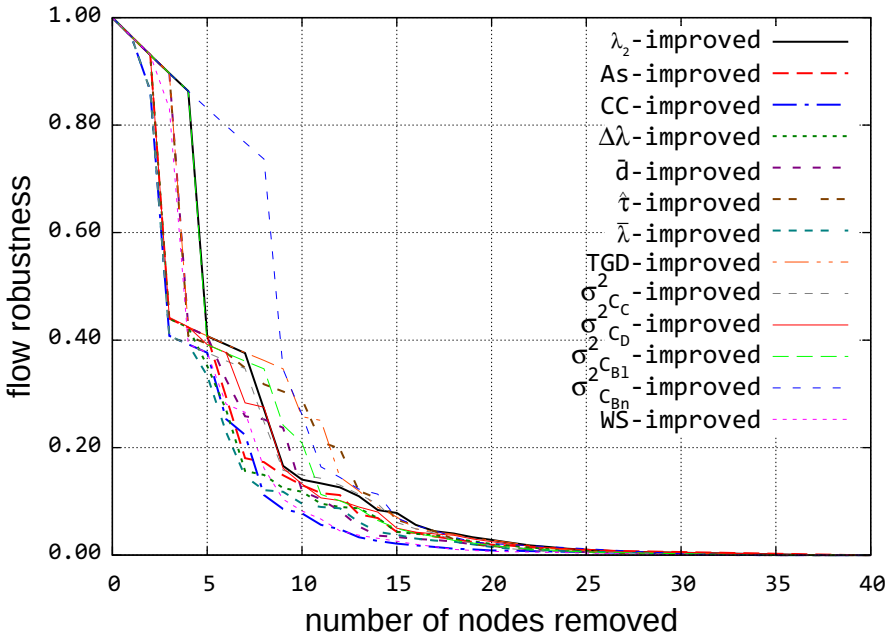

Figure B.68: Internet2 closeness attack

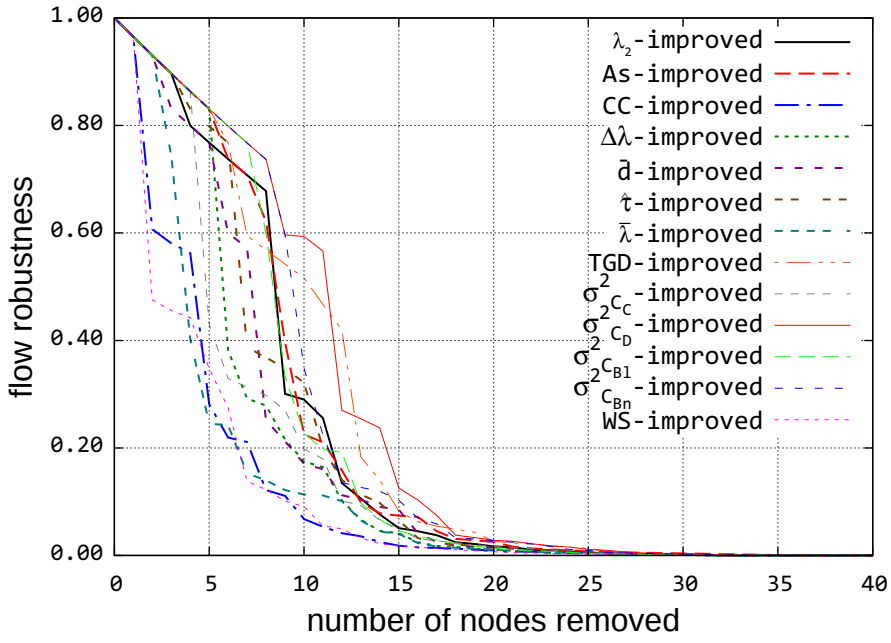

Figure B.69: Internet2 degree attack

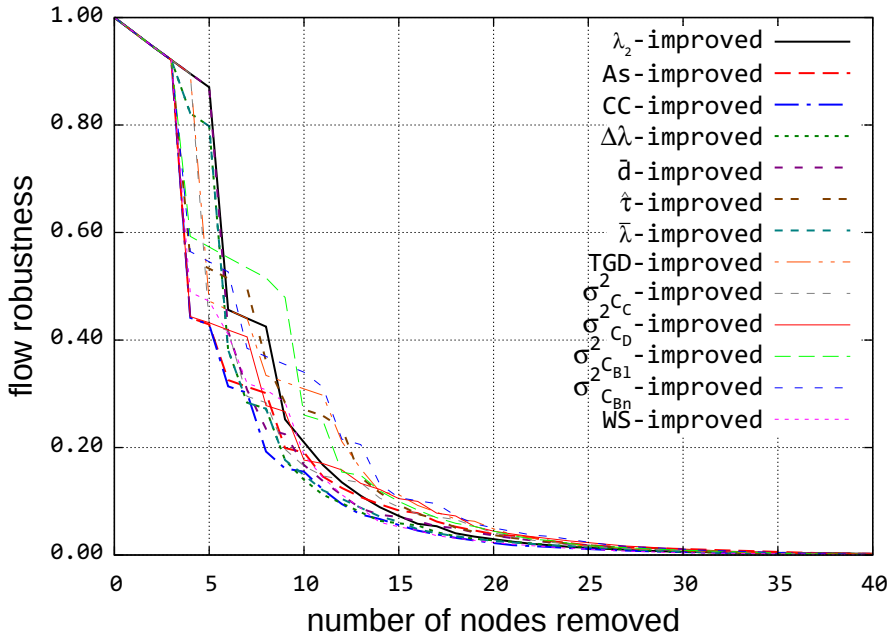

Figure B.70: CORONET betweenness attack

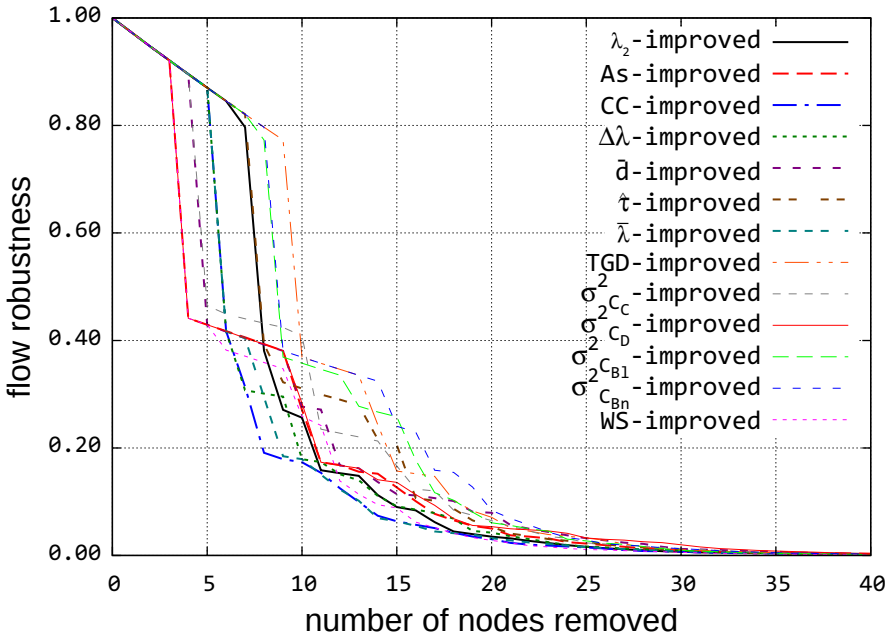

Figure B.71: CORONET closeness attack

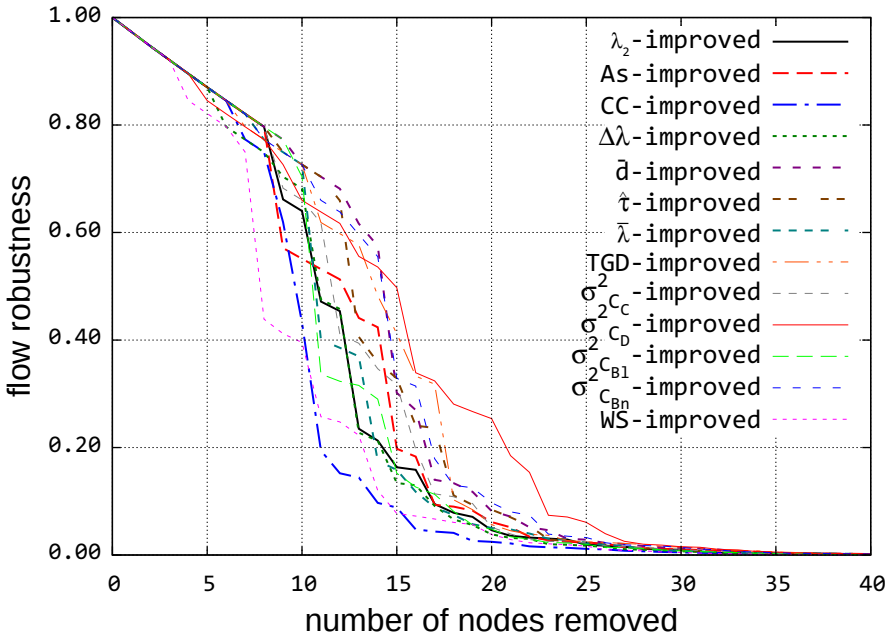

Figure B.72: CORONET-WP-L1 degree attack

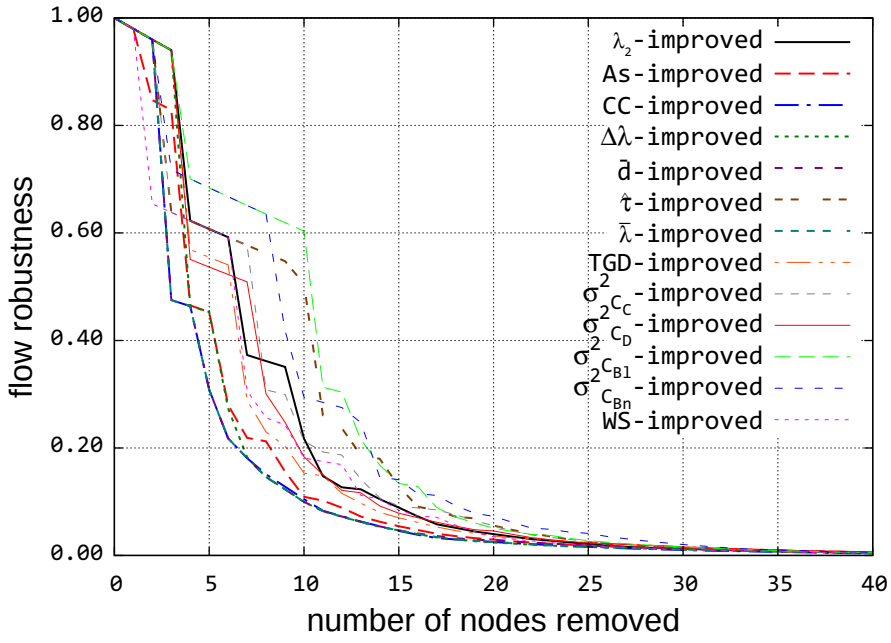

Figure B.73: Level 3 betweenness attack

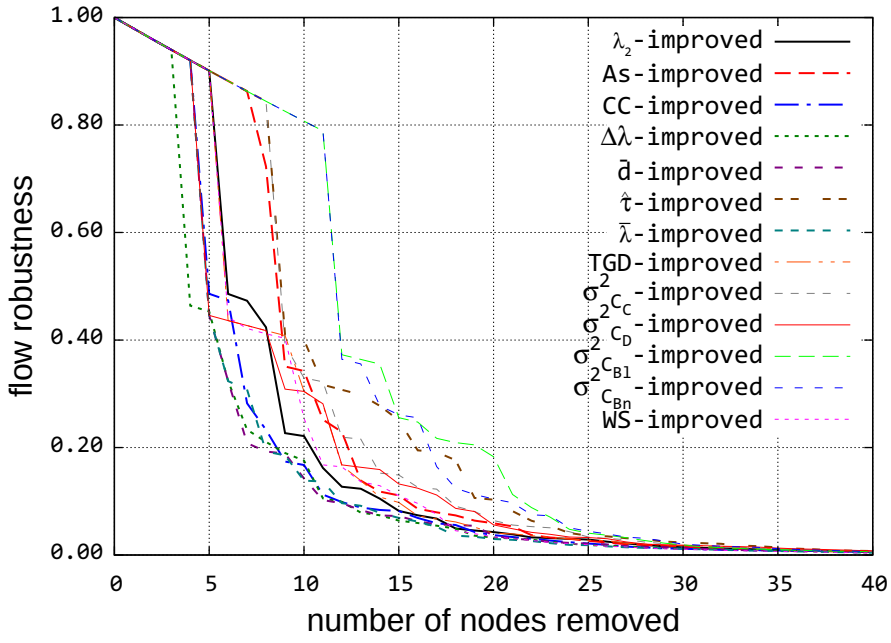

Figure B.74: Level 3 closeness attack

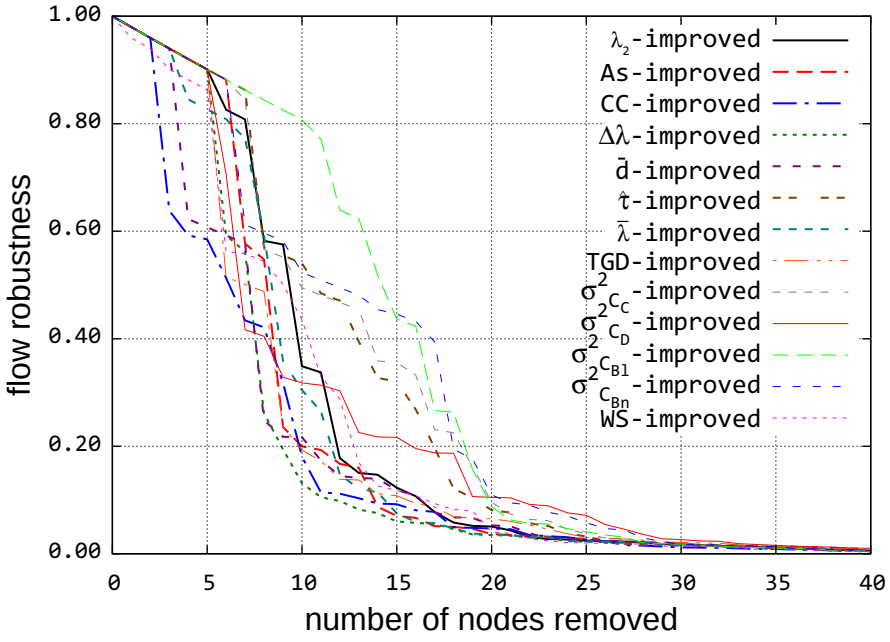

Figure B.75: Level 3 degree attack

B.5 Weighted Graphs Improvement Evaluation

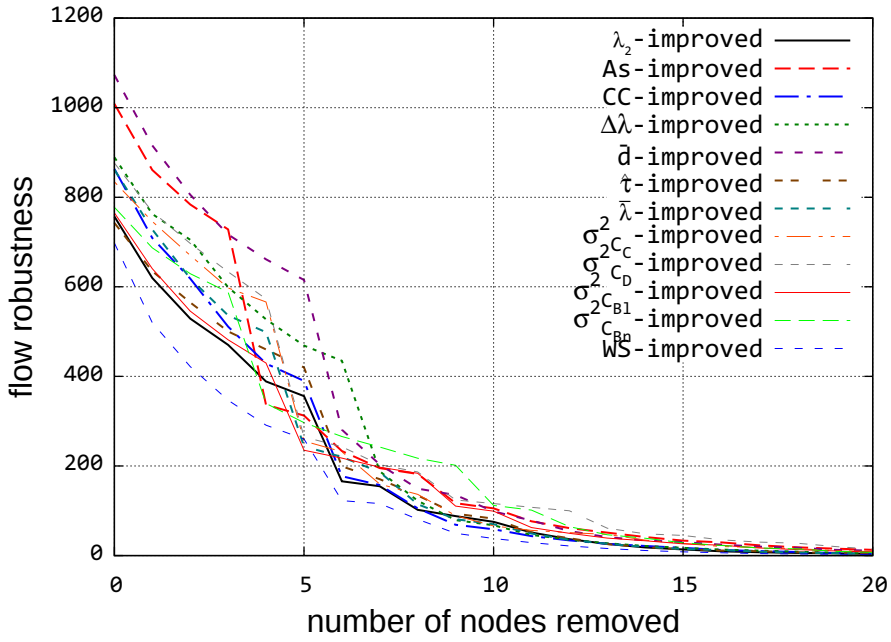

Figure B.76: CORONET betweenness attack

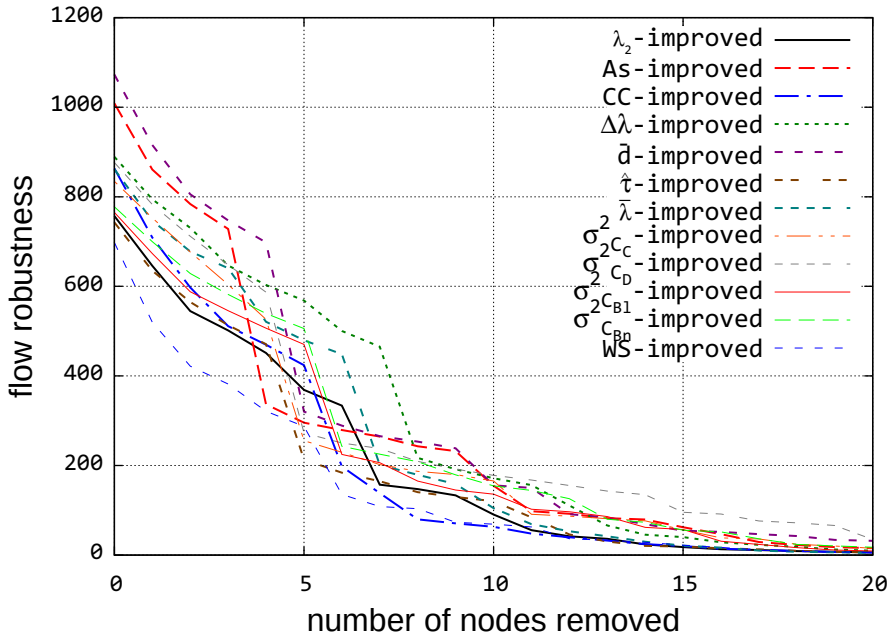

Figure B.77: CORONET closeness attack

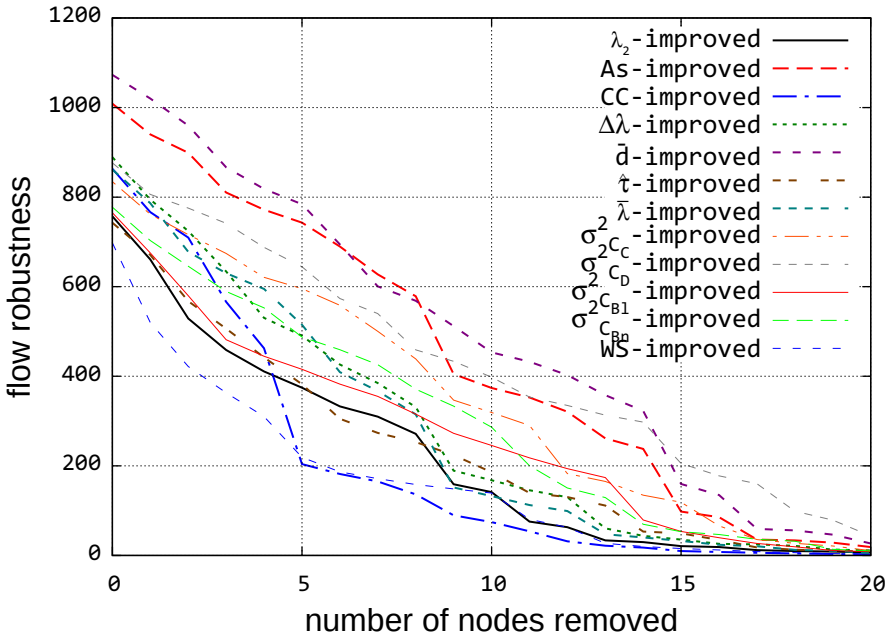

Figure B.78: CORONET degree attack

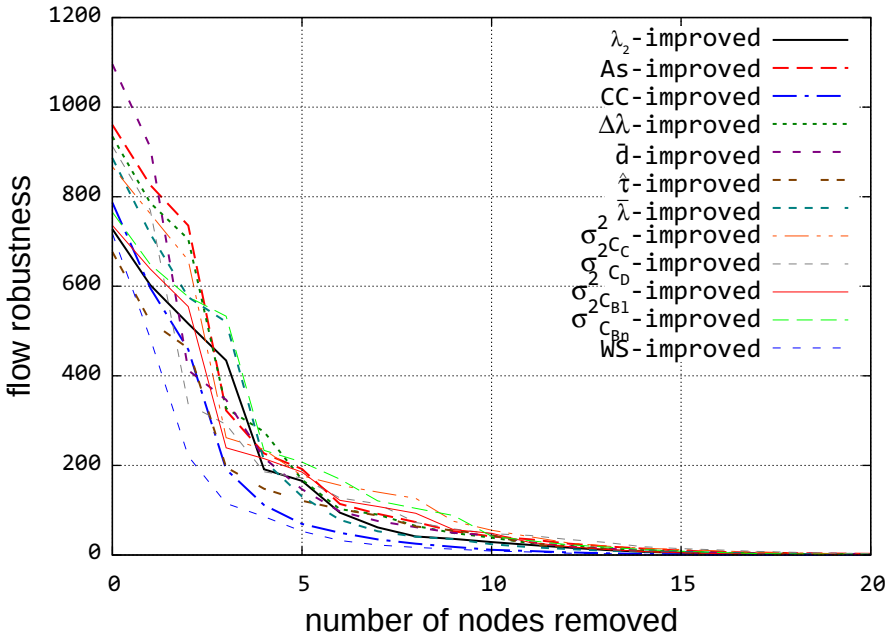

Figure B.79: Internet2 betweenness attack

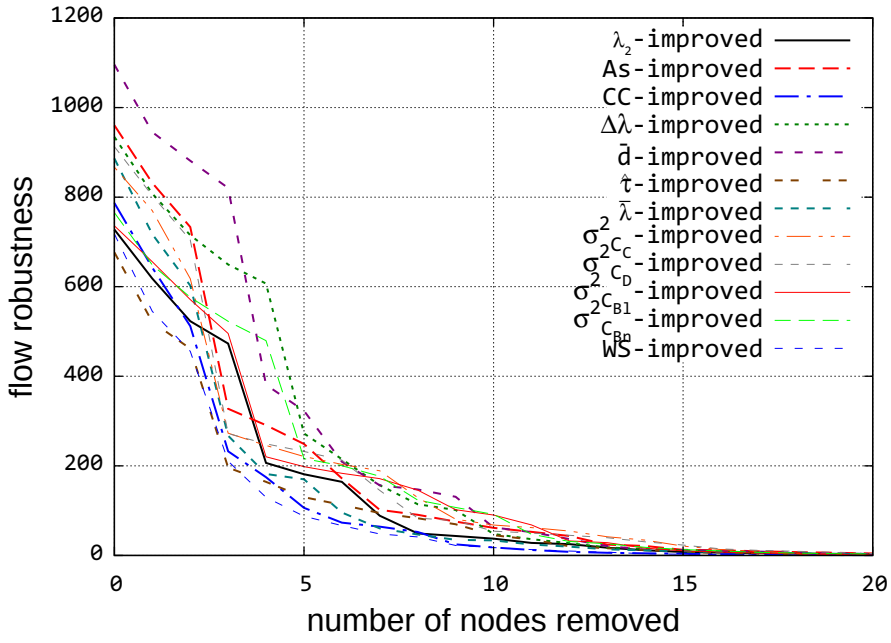

Figure B.80: Internet2 closeness attack

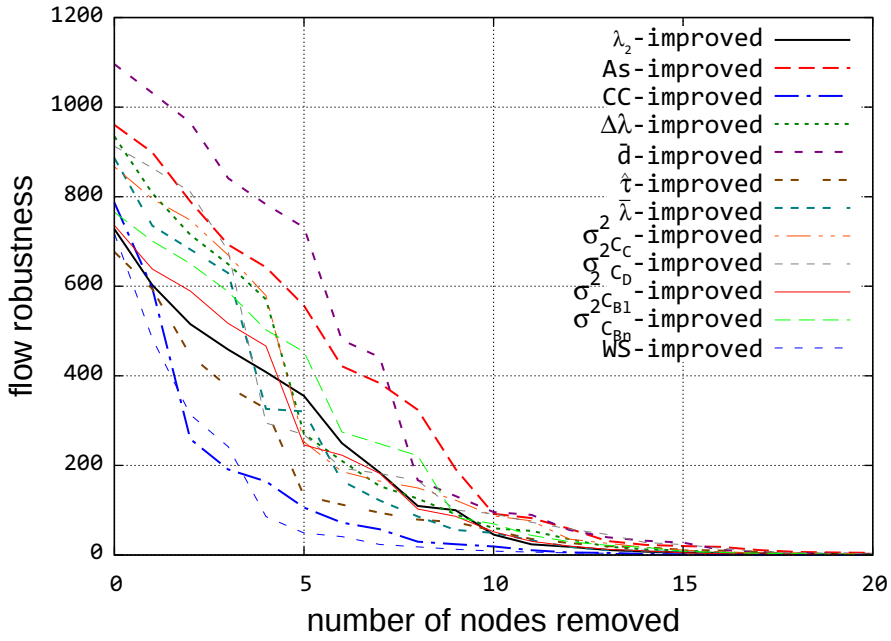

Figure B.81: Internet2 degree attack

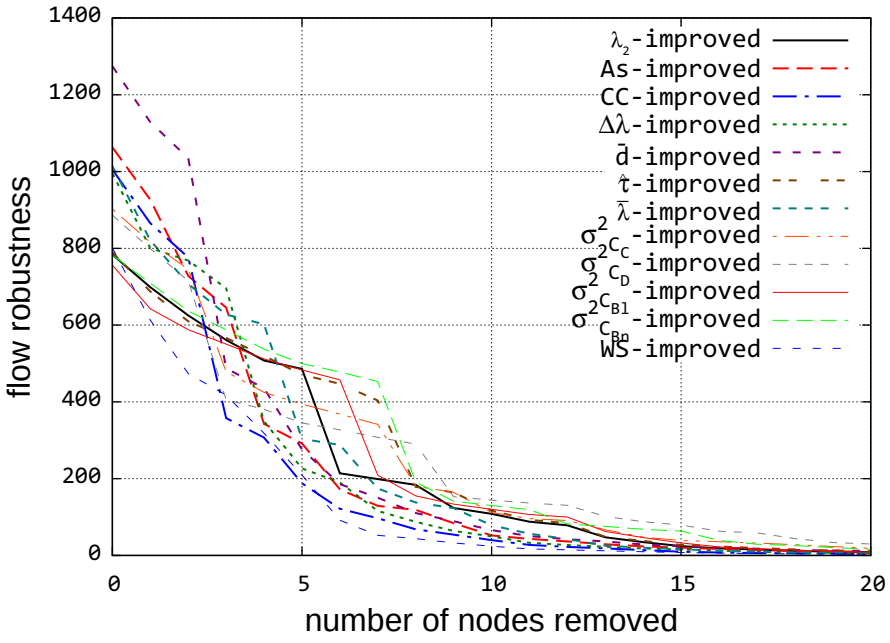

Figure B.82: Level 3 betweenness attack

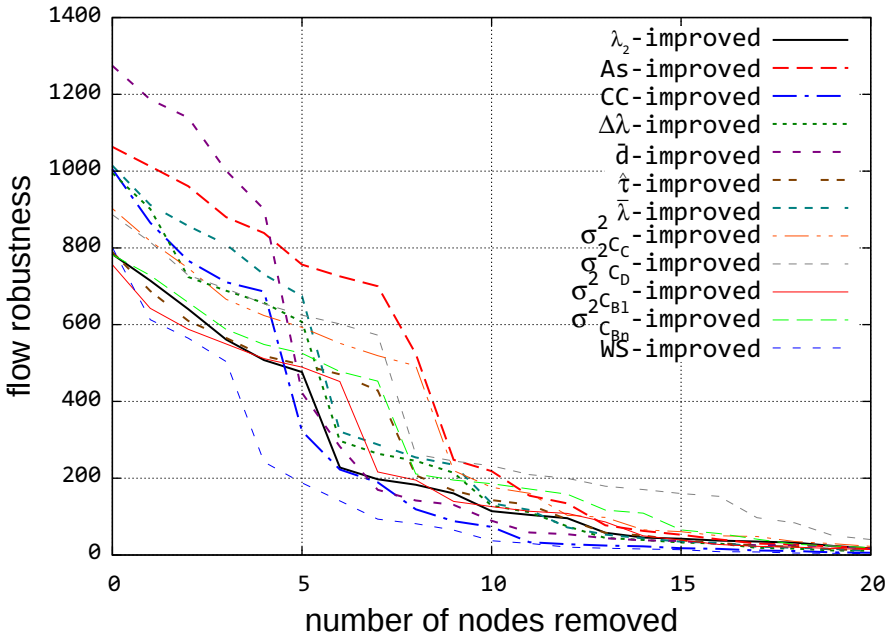

Figure B.83: Level 3 closeness attack

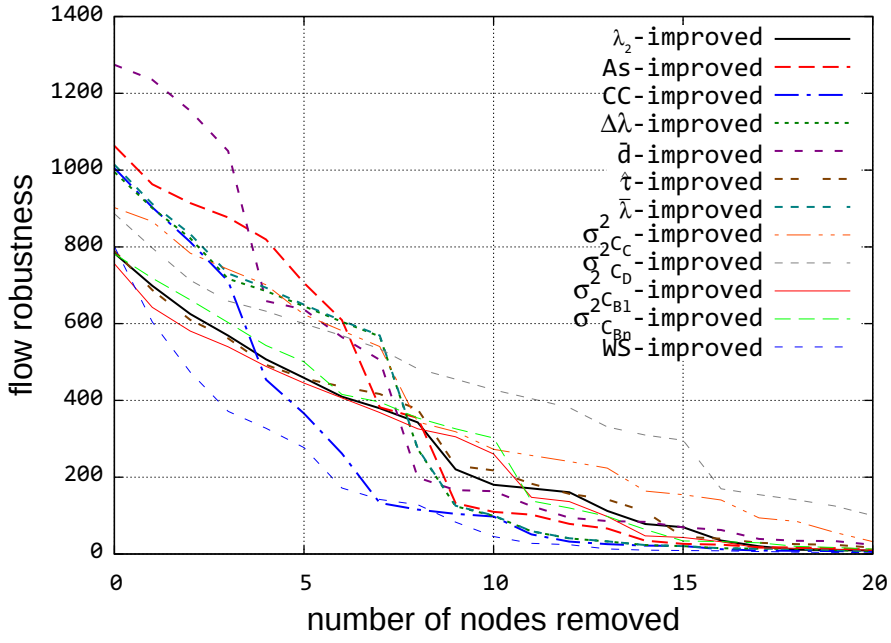

Figure B.84: Level 3 degree attack

B.6 Real-World Weighted Graphs Improvement Evaluation

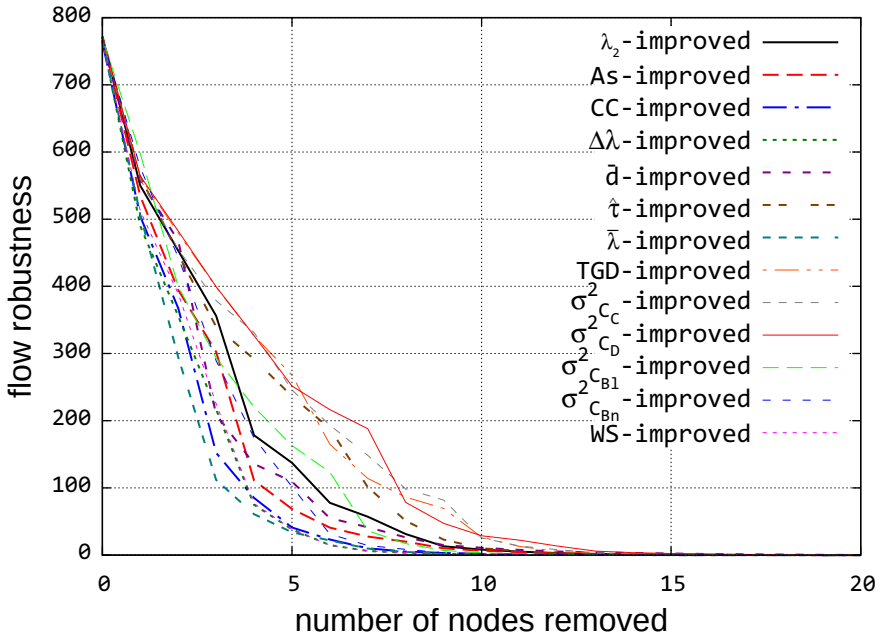

Figure B.85: GÉANT betweenness attack

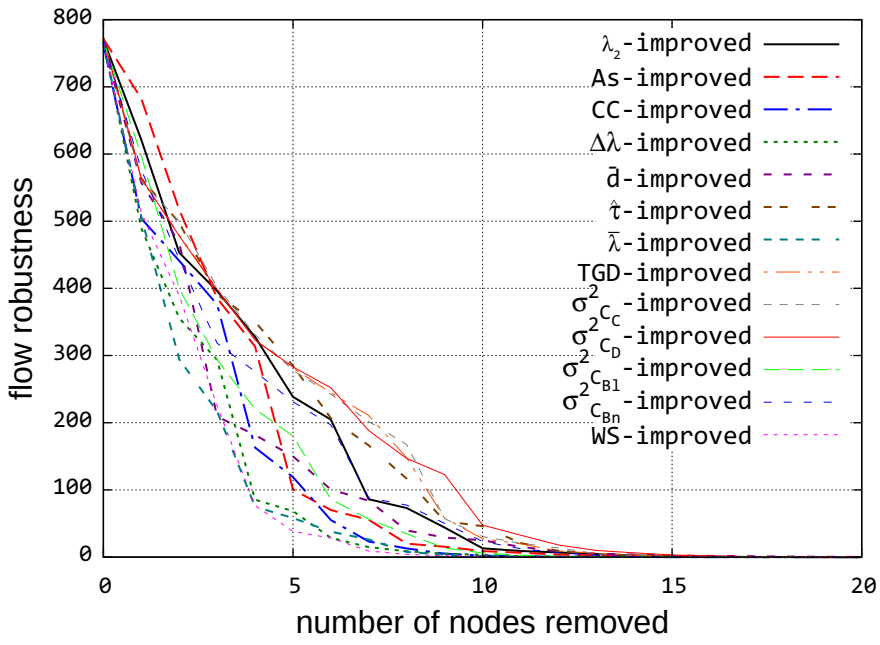

Figure B.86: GÉANT closeness attack

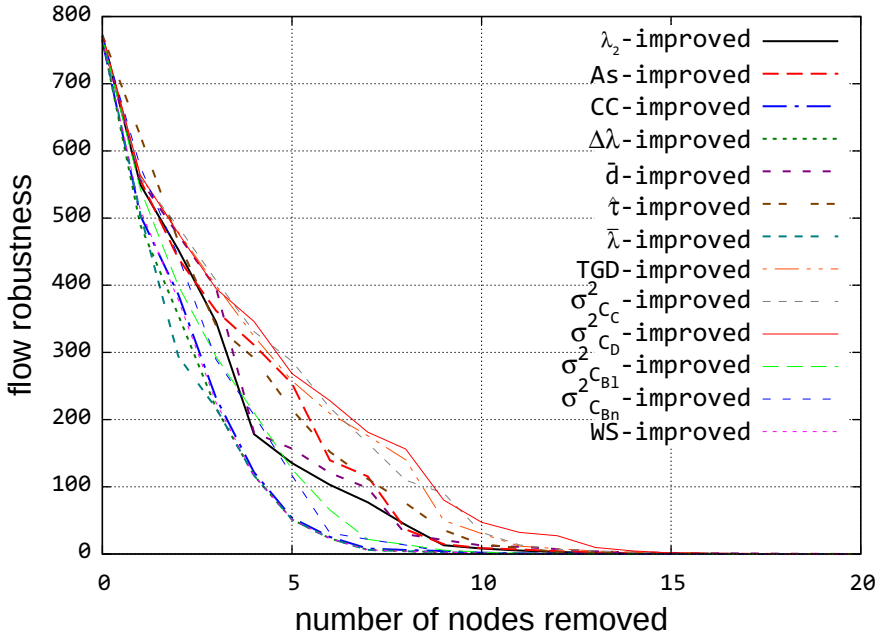

Figure B.87: GÉANT closeness attack

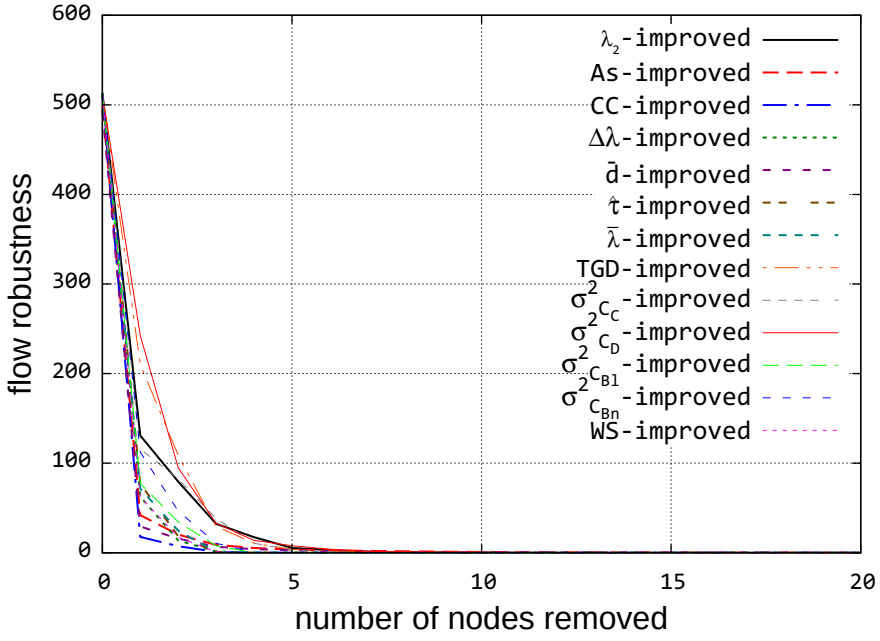

Figure B.88: CARNet betweenness attack

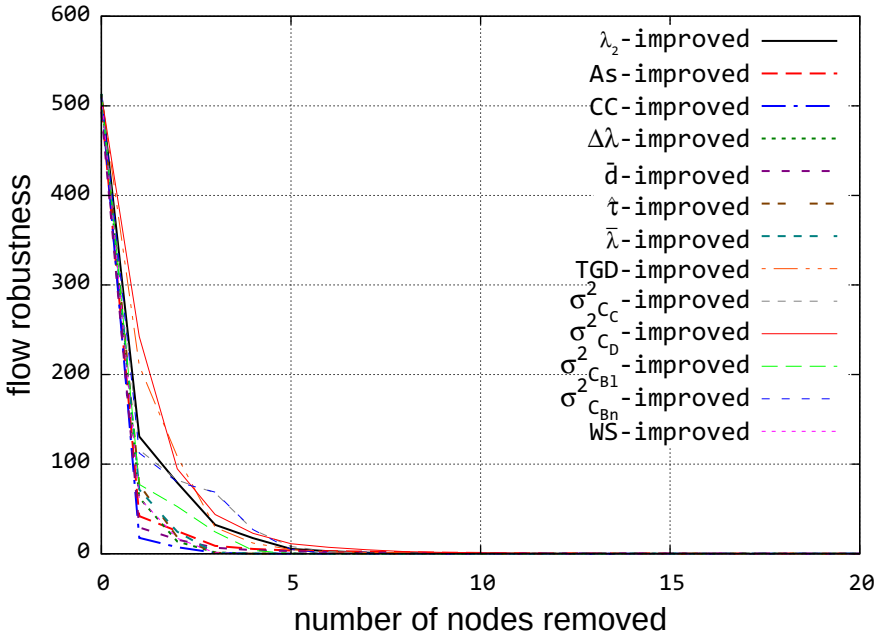

Figure B.89: CARNet closeness attack

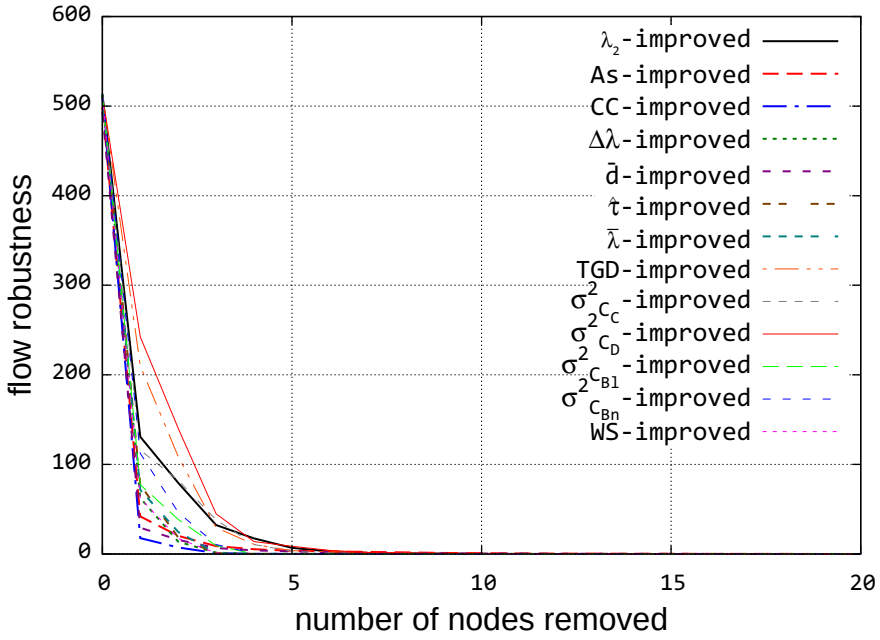

Figure B.90: CARNet closeness attack

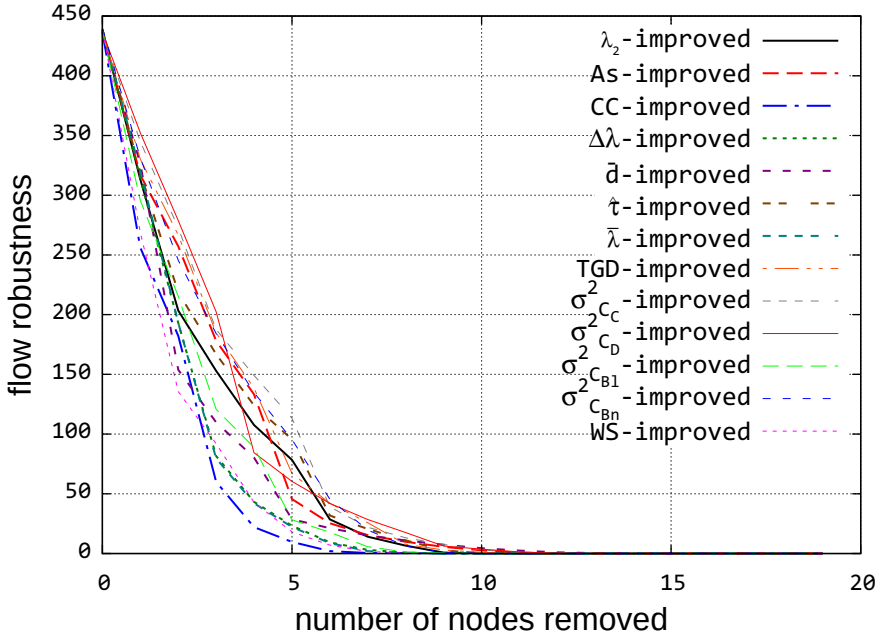

Figure B.91: InternetMCI betweenness attack

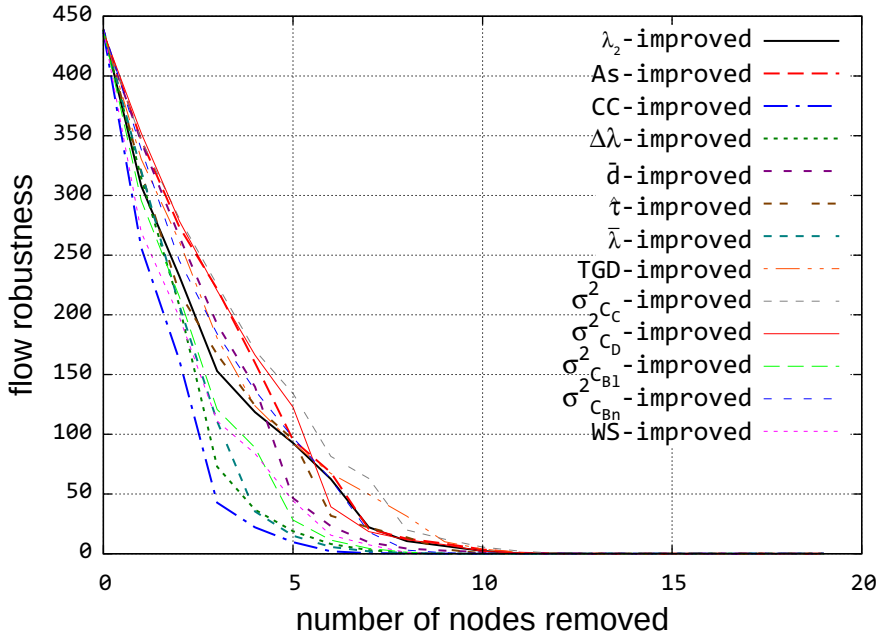

Figure B.92: InternetMCI closeness attack

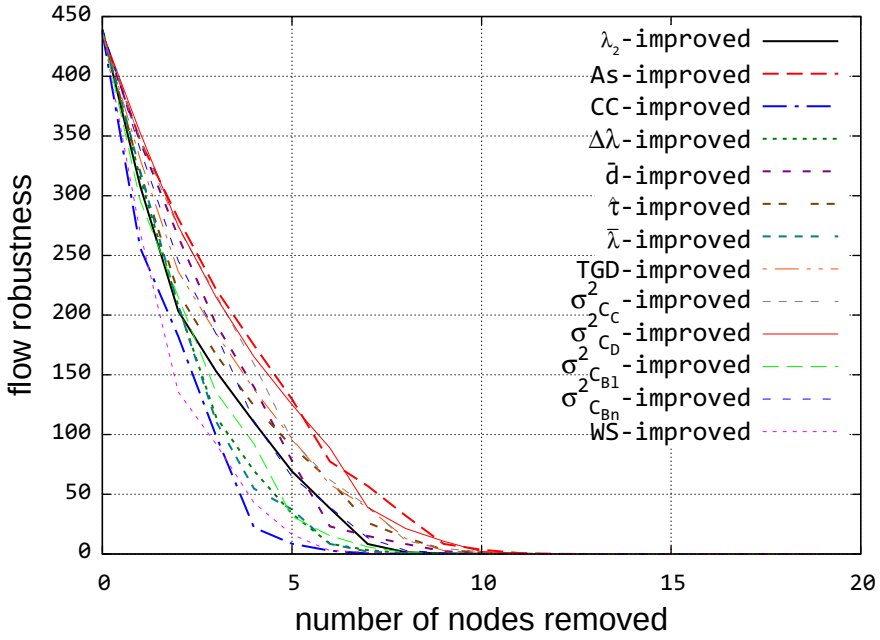

Figure B.93: InternetMCI closeness attack